1

WRDC-TR-90-8007
Volume V
Part 3

INTEGRATED INFORMATION SUPPORT SYSTEM (IISS)
Volume V - Common Data Model Subsystem
Part 3 - CDM1: IDEF1 Model of the CDM -- CDM Development Specification

J. Althoff, M. Apicella, S. Singh

Control Data Corporation
Integration Technology Services
2970 Presidential Drive
Fairborn, OH 45324-6209

September 1990

Final Report for Period 1 April 1987 - 31 December 1990

MANUFACTURING TECHNOLOGY DIRECTORATE
WRIGHT RESEARCH AND DEVELOPMENT CENTER
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433-6533

David Judson   25 July 91

DAVID L. JUDSON, Project Manager
WRDC/MTI
Wright-Patterson AFB, OH 45433-6533

DATE

FOR THE COMMANDER:

Bruce Rasmussen   25 July 91

BRUCE A. RASMUSSEN, Chief
WRDC/MTI
Wright-Patterson AFB, OH 45433-6533

DATE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for Public Release; Distribution is Unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) TBM620341000	5. MONITORING ORGANIZATION REPORT NUMBER(S) WRDC-TR- 90-8007 Vol. V, Part 3
6a. NAME OF PERFORMING ORGANIZATION Control Data Corporation; Integration Technology Services — 6b. OFFICE SYMBOL *(if applicable)*	7a. NAME OF MONITORING ORGANIZATION WRDC/MTI
6c. ADDRESS *(City,State, and ZIP Code)* 2970 Presidential Drive Fairborn, OH 45324-6209	7b. ADDRESS *(City, State, and ZIP Code)* WPAFB, OH 45433-6533
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Wright Research and Development Center, Air Force Systems Commands, USAF — 8b. OFFICE SYMBOL *(if applicable)* WRDC/MTI	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUM. F33600-87-C-0464
8c. ADDRESS *(City, State, and ZIP Code)* Wright-Patterson AFB, Ohio 45433-6533	10. SOURCE OF FUNDING NOS.
11. TITLE *(Include Security Classification)* See Block 19	

10. PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.
78011F	595600	F95600	20950607

12. PERSONAL AUTHOR(S)
Control Data Corporation: Althoff, J. L., Apicella, M. L., Singh, S.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT *(Yr.,Mo.,Day)*	15. PAGE COUNT
Final Report	4/1/87-12/31/90	1990 September 30	75

16. SUPPLEMENTARY NOTATION

WRDC/MTI Project Priority 6203

17. COSATI CODES			18. SUBJECT TERMS *(Continue on reverse if necessary and identify block no.)*
FIELD	GROUP	SUB GR.	
1308	0905		

19. ABSTRACT *(Continue on reverse if necessary and identify block number)*

This document provides a complete IDEF1 model of the Common Data (CDM) subsystem.

Block 11 - INTEGRATED INFORMATION SUPPORT SYSTEM (IISS)
Vol V - Common Data Model Subsystem
Part 3 - CDM1: IDEF1 Model of the CDM -- CDM Development Specification

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION	
UNCLASSIFIED/UNLIMITED x SAME AS RPT. DTIC USERS	Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE NO. *(Include Area Code)*	22c. OFFICE SYMBOL
David L. Judson	(513) 255-7371	WRDC/MTI

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

FOREWORD

This technical report covers work performed under Air Force Contract F33600-87-C-0464, DAPro Project. This contract is sponsored by the Manufacturing Technology Directorate, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio. It was administered under the technical direction of Mr. Bruce A. Rasmussen, Branch Chief, Integration Technology Division, Manufacturing Technology Directorate, through Mr. David L. Judson, Project Manager. The Prime Contractor was Integration Technology Services, Software Programs Division, of the Control Data Corporation, Dayton, Ohio, under the direction of Mr. W. A. Osborne. The DAPro Project Manager for Control Data Corporation was Mr. Jimmy P. Maxwell.

The DAPro project was created to continue the development, test, and demonstration of the Integrated Information Support System (IISS). The IISS technology work comprises enhancements to IISS software and the establishment and operation of IISS test bed hardware and communications for developers and users.

The following list names the Control Data Corporation subcontractors and their contributing activities:

SUBCONTRACTOR	ROLE
Control Data Corporation	Responsible for the overall Common Data Model design development and implementation, IISS integration and test, and technology transfer of IISS.
D. Appleton Company	Responsible for providing software information services for the Common Data Model and IDEF1X integration methodology.
ONTEK	Responsible for defining and testing a representative integrated system base in Artificial Intelligence techniques to establish fitness for use.
Simpact Corporation	Responsible for Communication development.
Structural Dynamics Research Corporation	Responsible for User Interfaces, Virtual Terminal Interface,and Network Transaction Manager design, development, implementation, and support.

Arizona State University	Responsible for test bed operations and support.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

SECTION 1

SCOPE

1.1 Identification

This document contains an IDEF1 data model of the information associated with the "AS-IS" Common Data Model (CDM) Subsystem of the Integrated Information Support System (IISS). Section 1 gives a brief overview of the CDM Subsystem. Section 2 identifies the other associated ICAM documents. The CDM1 model is depicted in a series of Function View diagrams shown in Section 3. Definitions and indexes for entities and attributes are contained in Sections 4 and 5 respectively.

1.2 Overview

The purpose of the model is to serve as a guide for the development of the CDM Database and the CDM Processor (CDMP). These components of the CDM Subsystem are described in the following paragraphs. However, not all aspects of the CDM1 model are implemented by the current IISS development specifications and demonstration software.

The CDM Database

The CDM database is the data dictionary of the IISS. It captures knowledge of the locations, characteristics, and interrelationships of all shared data in the system. The most significant feature of the CDM database is that it implements the ANSI/X3/SPARC concepts of the three-schema approach to data management. The three types of schemas are the conceptual schema (CS), the internal schemas (IS), and the external schemas (ES).

The conceptual schema describes a neutral, integrated view of the shared data resource. There is one conceptual schema in an enterprise. It is independent of physical database structures and boundaries and is neutral to biases of individual applica- tions. Each external schema represents a user or application view of data. Each internal schema represents the logical structure of a local DBMS whether hierarchical, network, or relational.

The CDM database, itself is implemented as a relational database, which presently resides on a VAX 11/780 computer. It is accessed by the CDMP at compile-time to generate appropriate local DBMS calls against internal schemas to process a user's NDML request against an external schema.

The CDM Processor

The CDMP is the distributed database manager of the IISS. It builds on top of local DBMS services to provide data access. The CDMP plays both a compile-time and a run-time role in the processing of transactions. The compile-time component is called the CDMP Precompiler. The run-time components are called the CDMP Distributed Request Supervisor (DRS) and the CDMP Aggregator.

The CDMP Precompiler performs the following functions for each data request:

1. Parses the request,
2. Transforms the request from an external schema access to a conceptual schema access,
3. Decomposes the request into subrequests, each of which accesses one internal schema,
4. Determines an appropriate access path for each subrequest and generates code that can be processed by the pertinent local DBMS,
5. Generates code to transform any data to be extracted from local databases from internal to conceptual schema format (this code is called a Request Processor Packet or RPP),
6. Generates code to transform any data results from conceptual to external schema format (this code is called a C/E Transformer or CEX), and
7. Generates code to invoke appropriate RPPs and CEXs at run-time, via calls to the NTM Subsystem.

The CDMP Precompiler accesses the CDM database to find metadata for the interschema transforms and integrity constraints for update requests.

After successful precompilation of a user's program, which contains imbedded data requests in an SQL-like language called the Neutral Definition Manipulation Language (NDML), the CDMP has produced the following code modules:

1. Modified user program, which now contains calls to the NTM, which will activate appropriate processes at run-time.
2. One Request Processor (RP) per DBMS that manages data to be accessed by the user program. Each RP contains one or more RPPs.
3. One Conceptual-to-External Transformer (CEX), which will deliver query results to the modified user program at run-time.

The CDMP Distributed Request Supervisor (DRS) has responsibility for scheduling and coordinating the various subrequests of user transactions. The DRS uses request graphs produced by the CDMP Precompiler to determine which operations are to be performed where. The DRS also uses knowledge of communications costs and intermediate result volumes in its algorithm for scheduling RPPs. Request Processors always deliver results as relations. The relations are operated upon by the Aggregators.

Aggregators are called to perform single functions, e.g., a union or a join, on two sets of data, each of which exists in a single sequential file. These data sets are the results of an RPP or processing by another Aggregator. An Aggregator always deals with data in conceptual schema format.

CDM1 Overview

CDM1 is a semantic data model of the IISS metadata, i.e. a semantic model of data about data. It was built using the IDEF1 data modeling approach with some minor extensions. (These extensions along with others, have now been refined and formalized in the IDEF1 - Extended Manual). Part of the metadata modeled by CDM includes the IISS Conceptual Schema View of manufacturing data, which is itself represented by an IDEF1 model.

The conceptual schema portion of the CDM1 model is related to portions that describe internal and external schemas. An internal schema describes a local database structure in just enough detail to give the CDMP adequate information to generate code that can be processed by the pertinent local DBMS. The mappings between the conceptual schema metadata and the internal schema metadata are not simple, because one of the requirements of the IISS is that it provide integration of data in existing databases. IISS does not have the luxury of supporting only certain clean database structures. It is very likely that an attribute may be represented by one or more data files, which may be in different databases and even on different computers, or represented by relationships between record types.

An external schema describes the portion of the conceptual schema that is within the purview of a user or application. An external schema that is equivalent to a view in the relational model. The conceptual-to-external schema mapping part of the CDM1 is straightforward. The present implementation of the CDM subsystem supports any external schema that can be formed by joining conceptual schema entities and selecting attributes.

Thus, the CDM1 model is a semantic data model that describes the logical structure of the CDM database. The CDM1 represents the conceptual schema, the internal schemas and their mappings from the conceptual schema, and the external schemas and their mappings from the conceptual schema.

SECTION 2

DOCUMENTS

2.1 Applicable Documents

Related ICAM Documents included:

UM620141001	CDM Administrator's Manual
PRM620141200	NDML Programmer's Reference Manual
UM620141100	Neutral Data Definition Language (NDDL) User's Guide
UM620141002	Information Modeling Manual - IDEF1 Extended
DS620141320	Data Aggreegator DS
DS620141310	Distributed Request Supervisor DS
DS620141200	NDML Precompiler DS

2.2 Terms and Abbreviations

APL	Attribute Pair List
AUC	Attribute Use Class
CDMP	Common Data Model Processor
CI	Configuration Item
CS	Conceptual Schema
DML	Data Manipulation Language
DRS	Distributed Request Supervisor (previously SS: Stager/Scheduler)
ES	External Schema
ICAM	Integrated Computer Aided Manufacturing
IS	Internal Schema
NDML	Neutral Data Manipulation Language
RP	Request Processor
RFT	Result Field Table
SDS	System Design Specification

SECTION 3

FUNCTION VIEW DIAGRAMS

The CDM1 Model is depicted in seventeen Function View Diagrams (FEO's) which are shown in Figures 3-1 through 3-18.

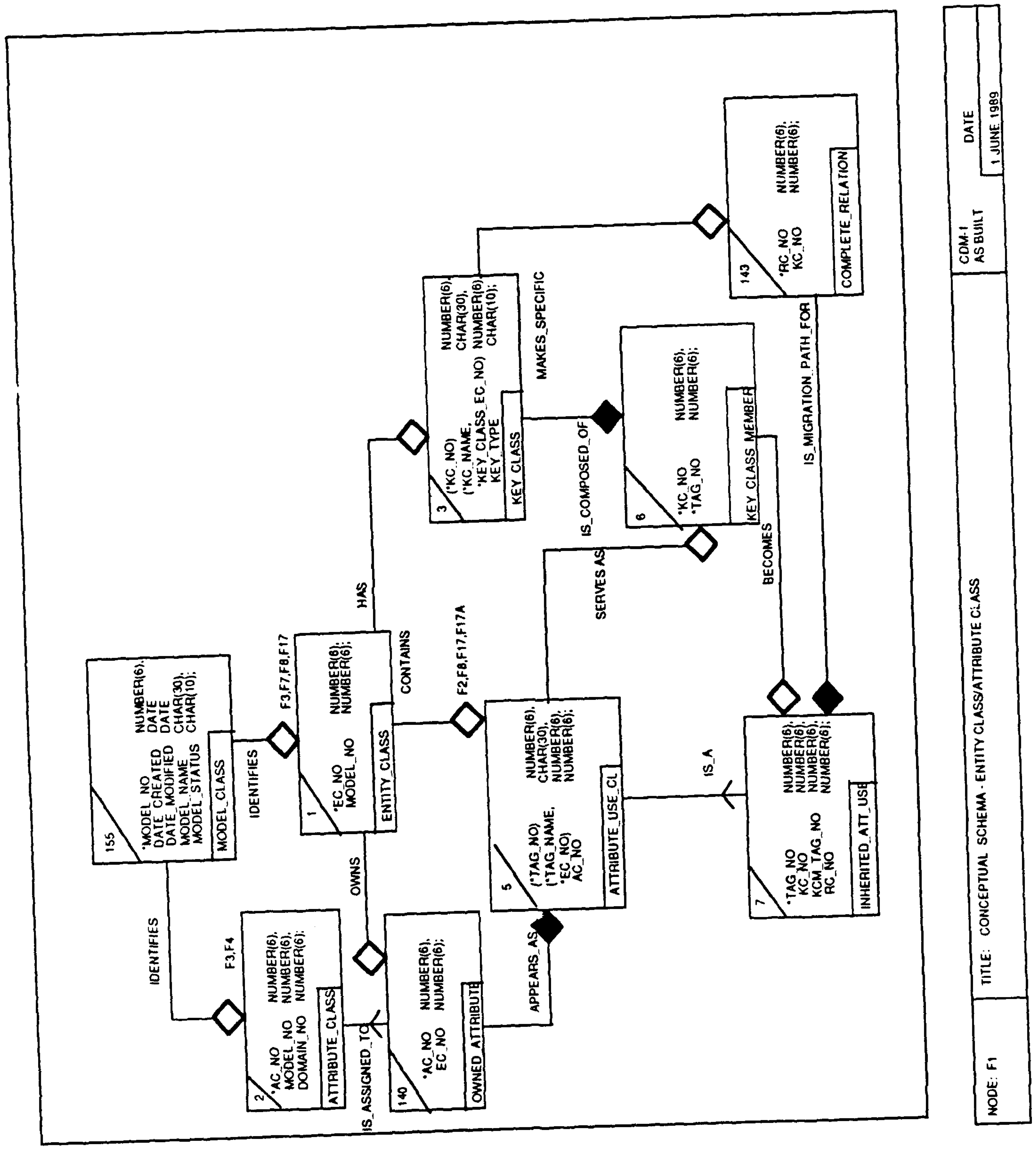

Figure 3-1. F1: Conceptual Schema - Entity Class/Attribute Class

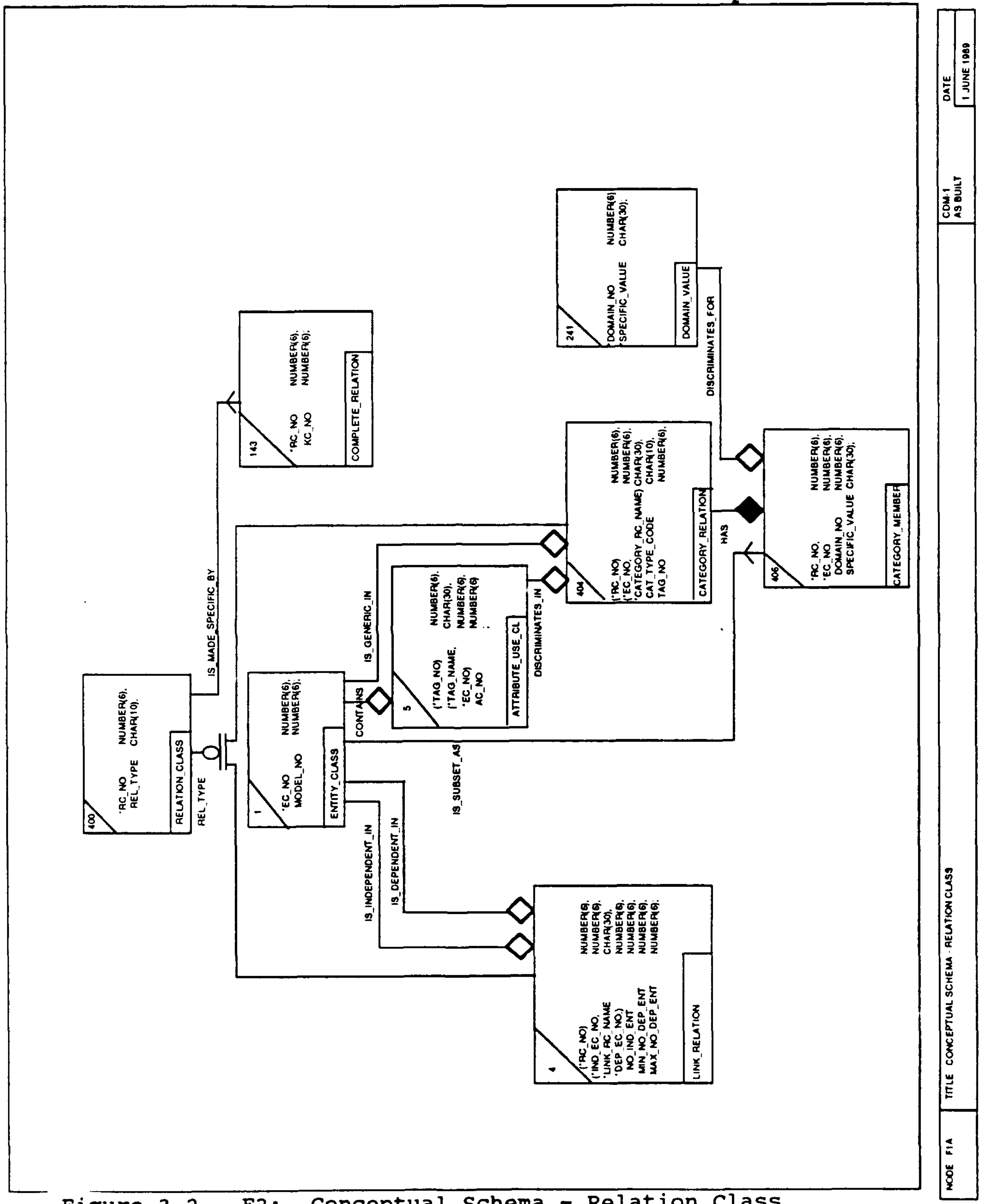

Figure 3-2. F2: Conceptual Schema - Relation Class

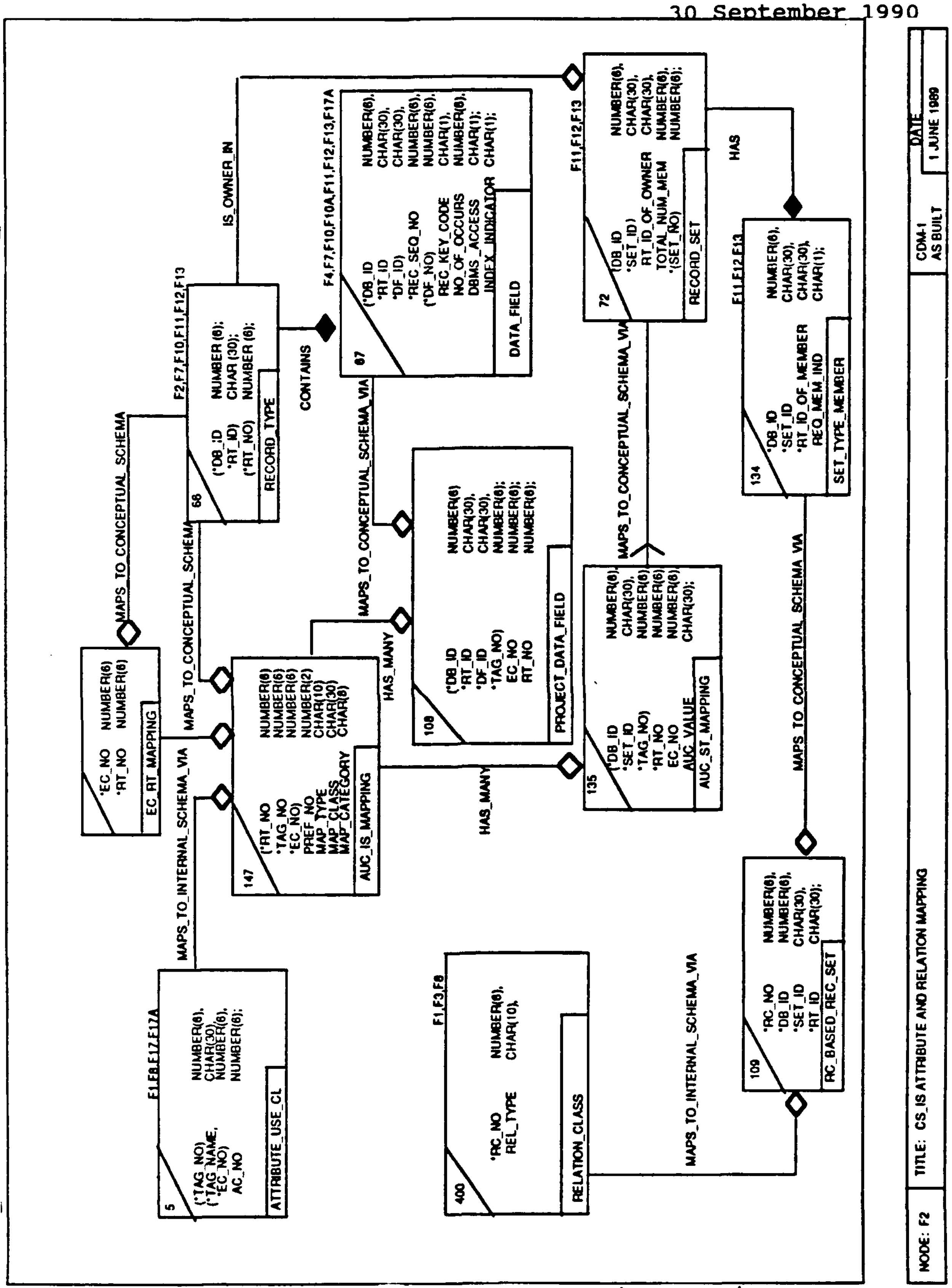

Figure 3-3. F3: CS-IS Attribute and Relation Mapping

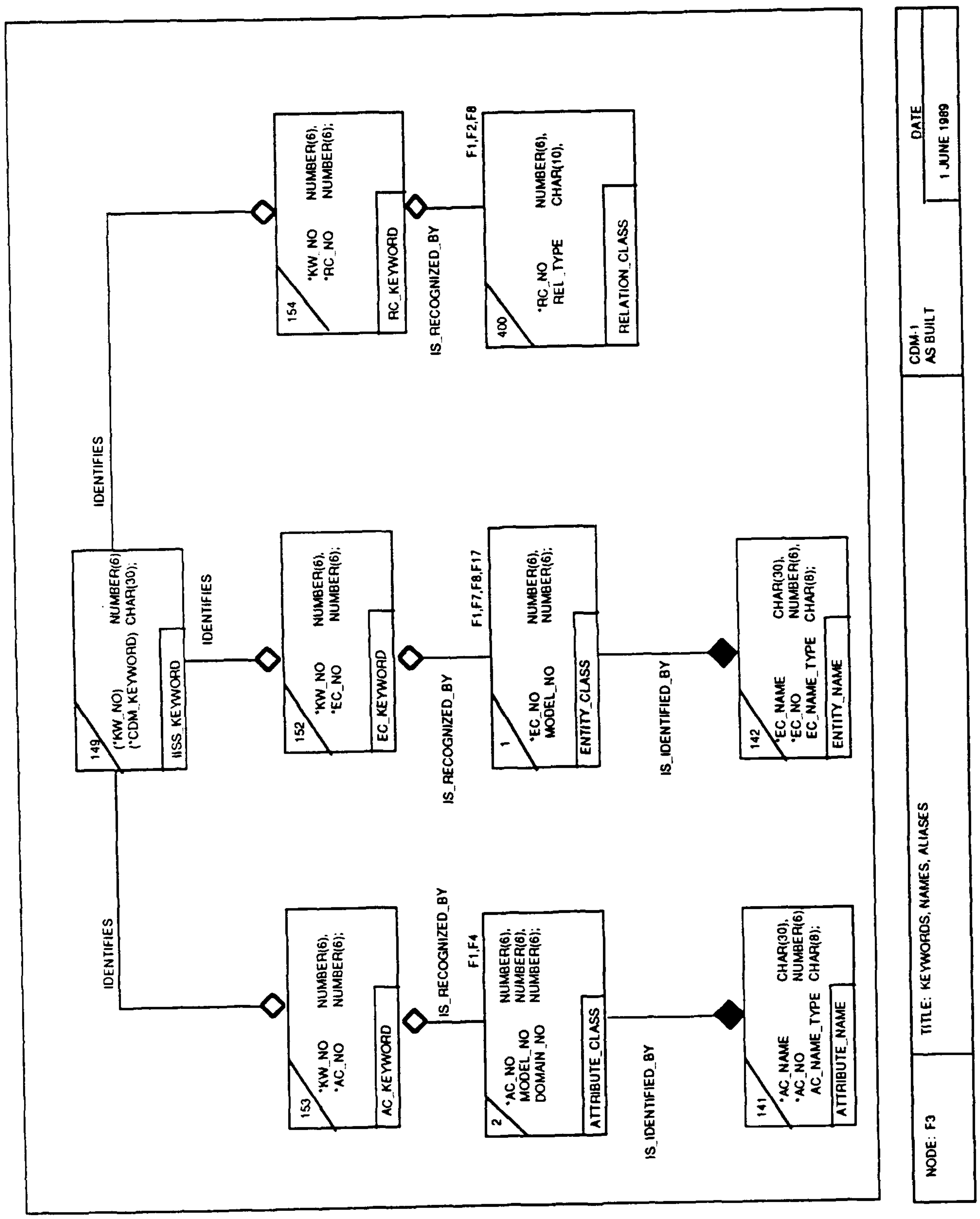

Figure 3-4. F4: Keywords, Names, Aliases

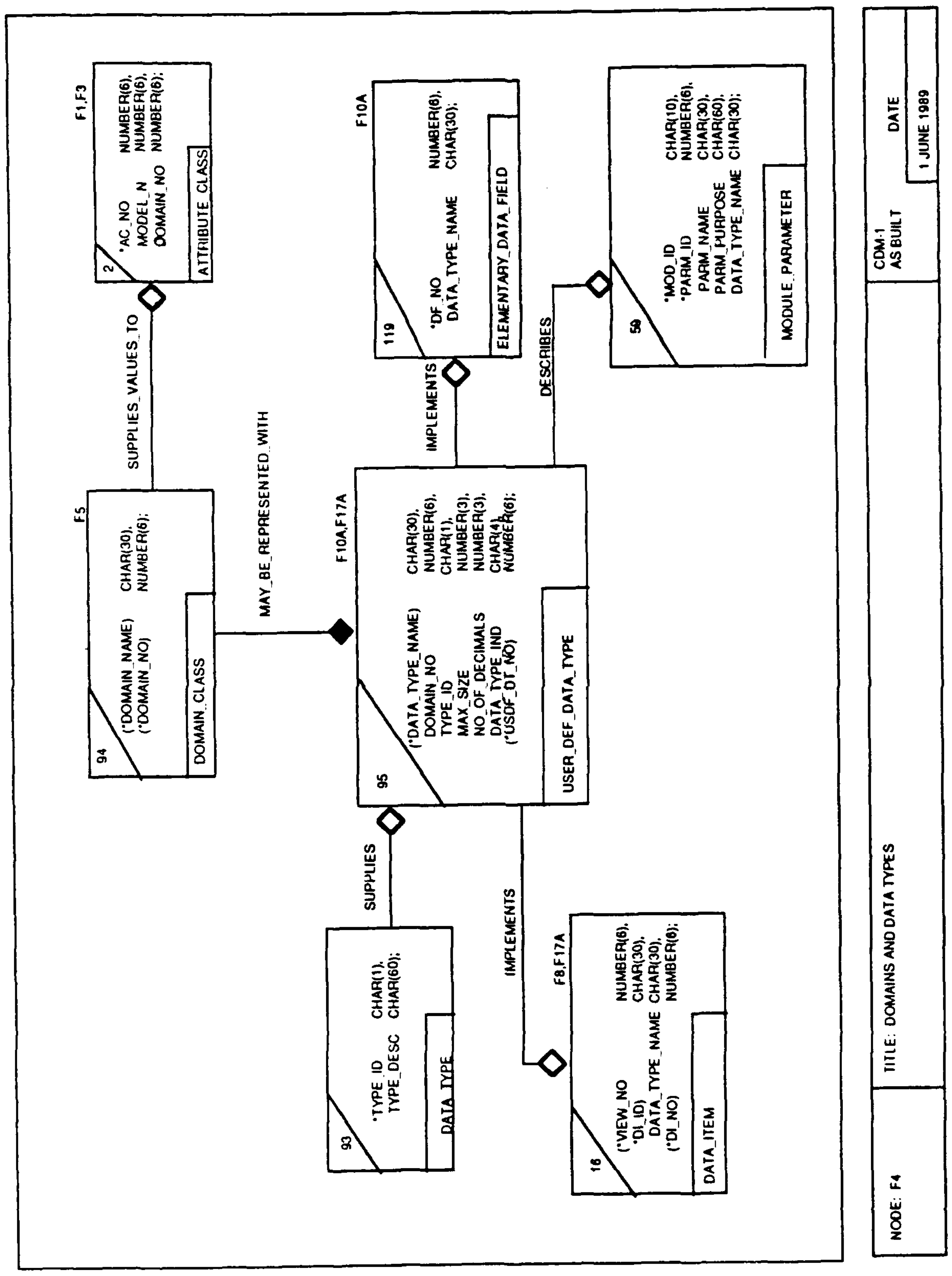

Figure 3-5. F5: Domains and Data Types

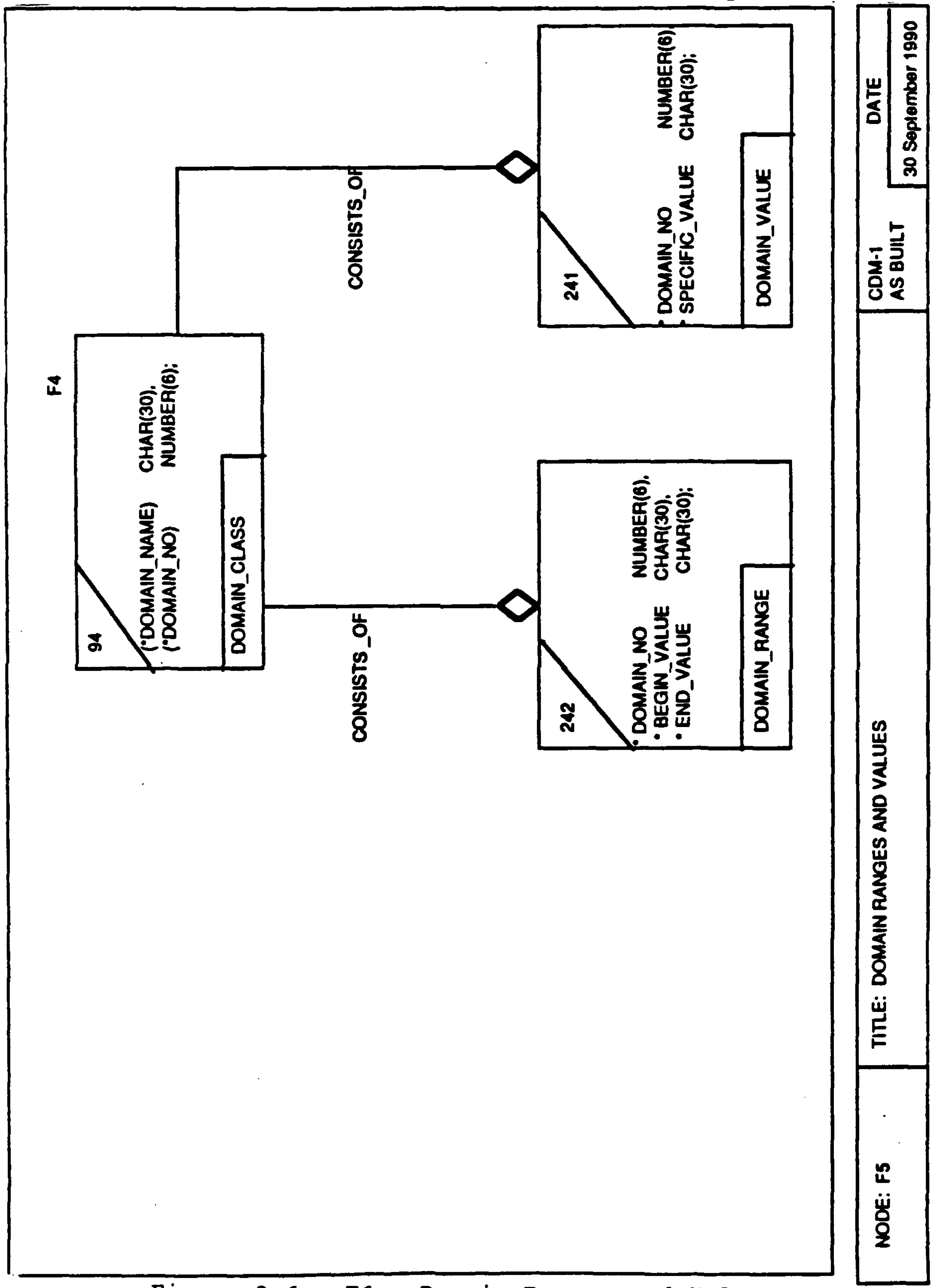

Figure 3-6. F6: Domain Ranges and Values

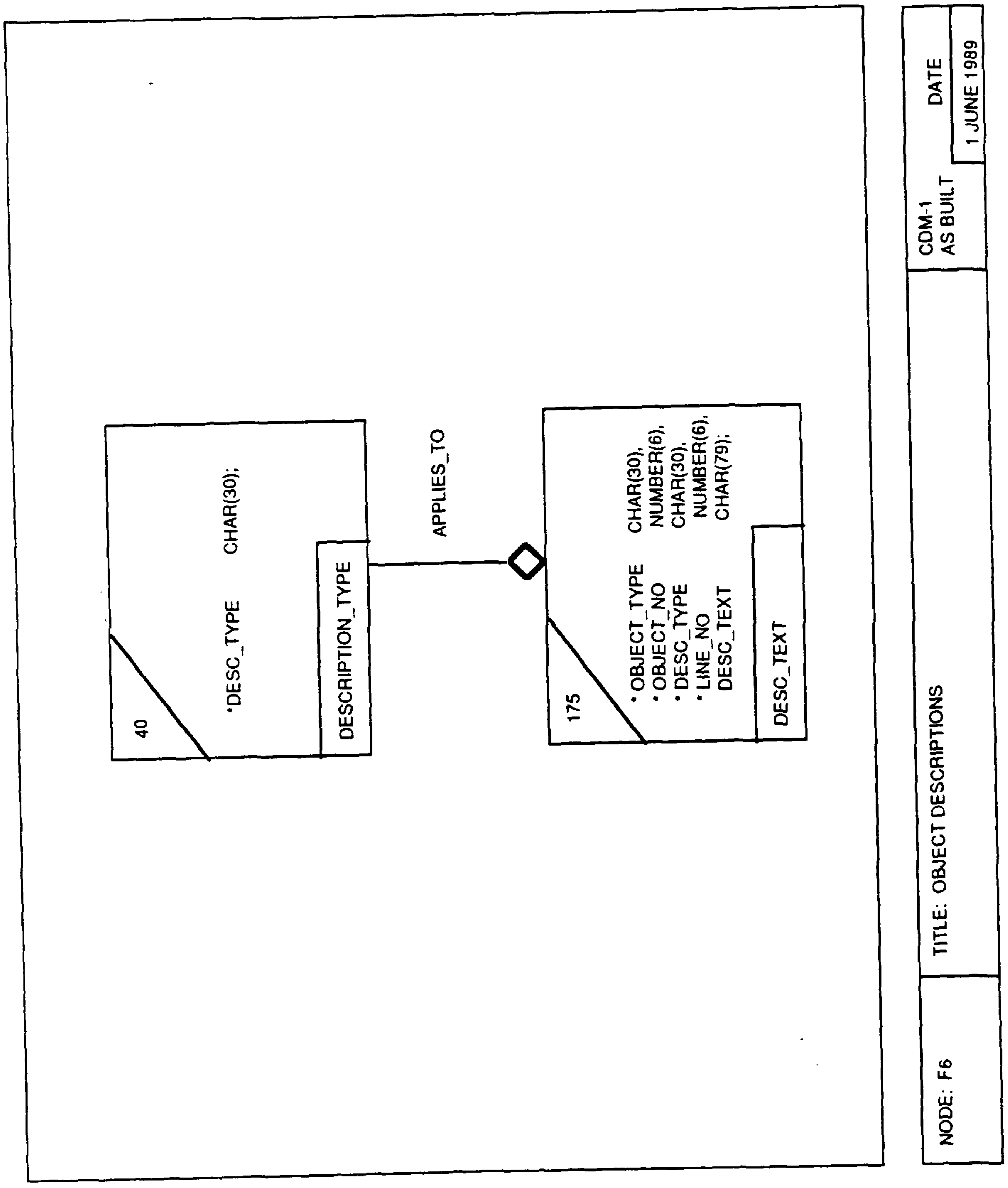

Figure 3-7. F7: Object Descriptions

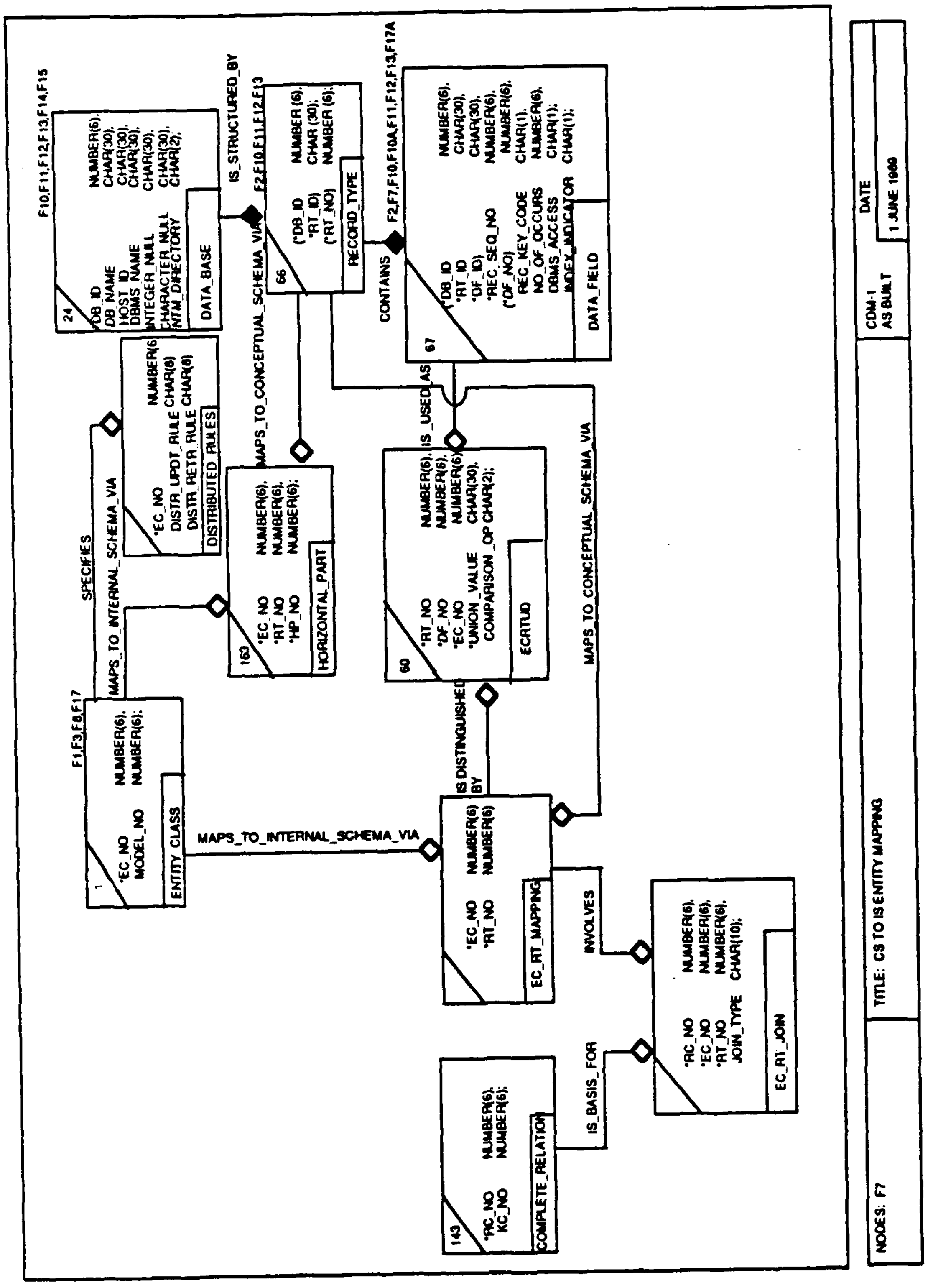

Figure 3-8. F8: CS-IS Entity Mapping

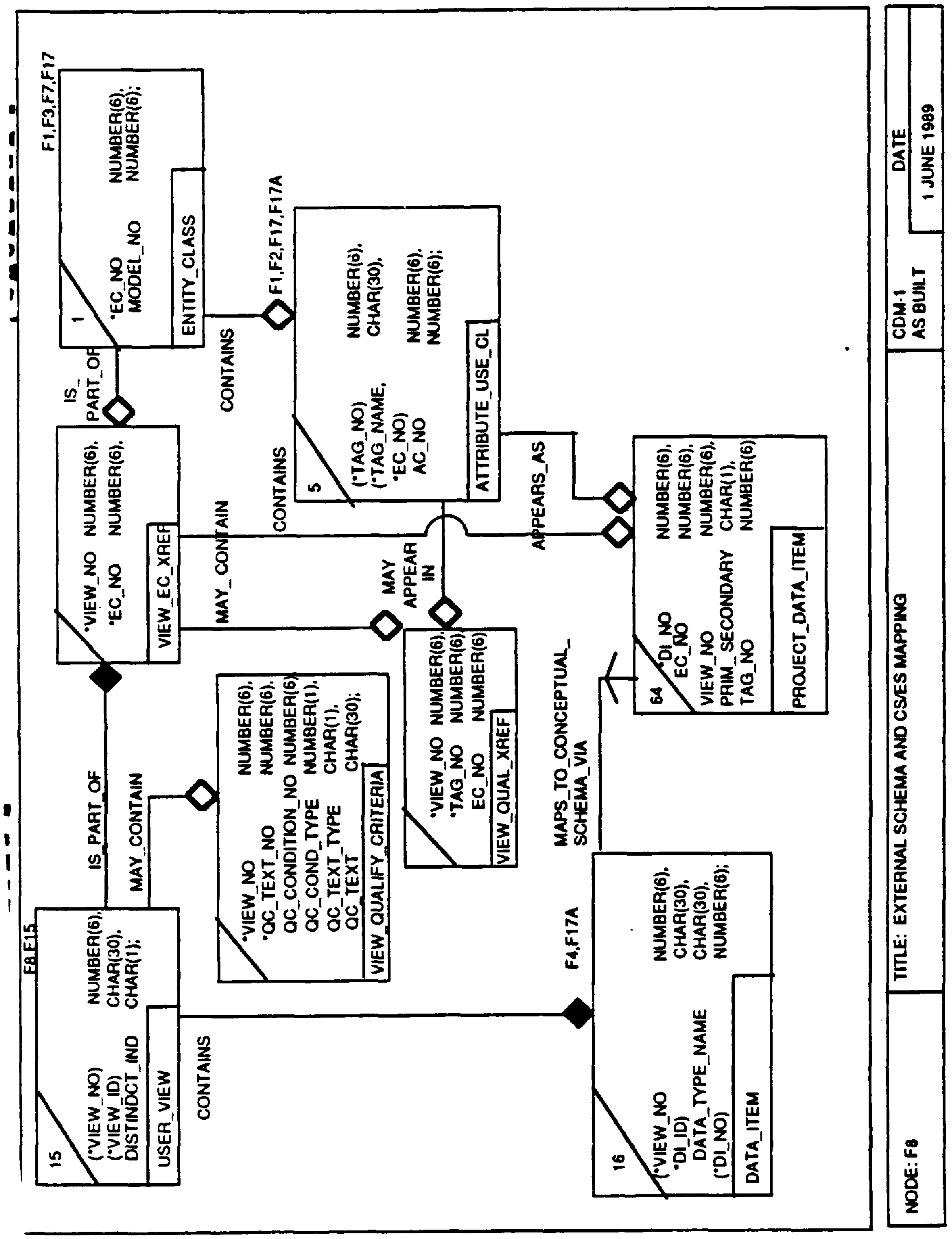

Figure 3-9. F9: External Schema and CS-IS Mapping

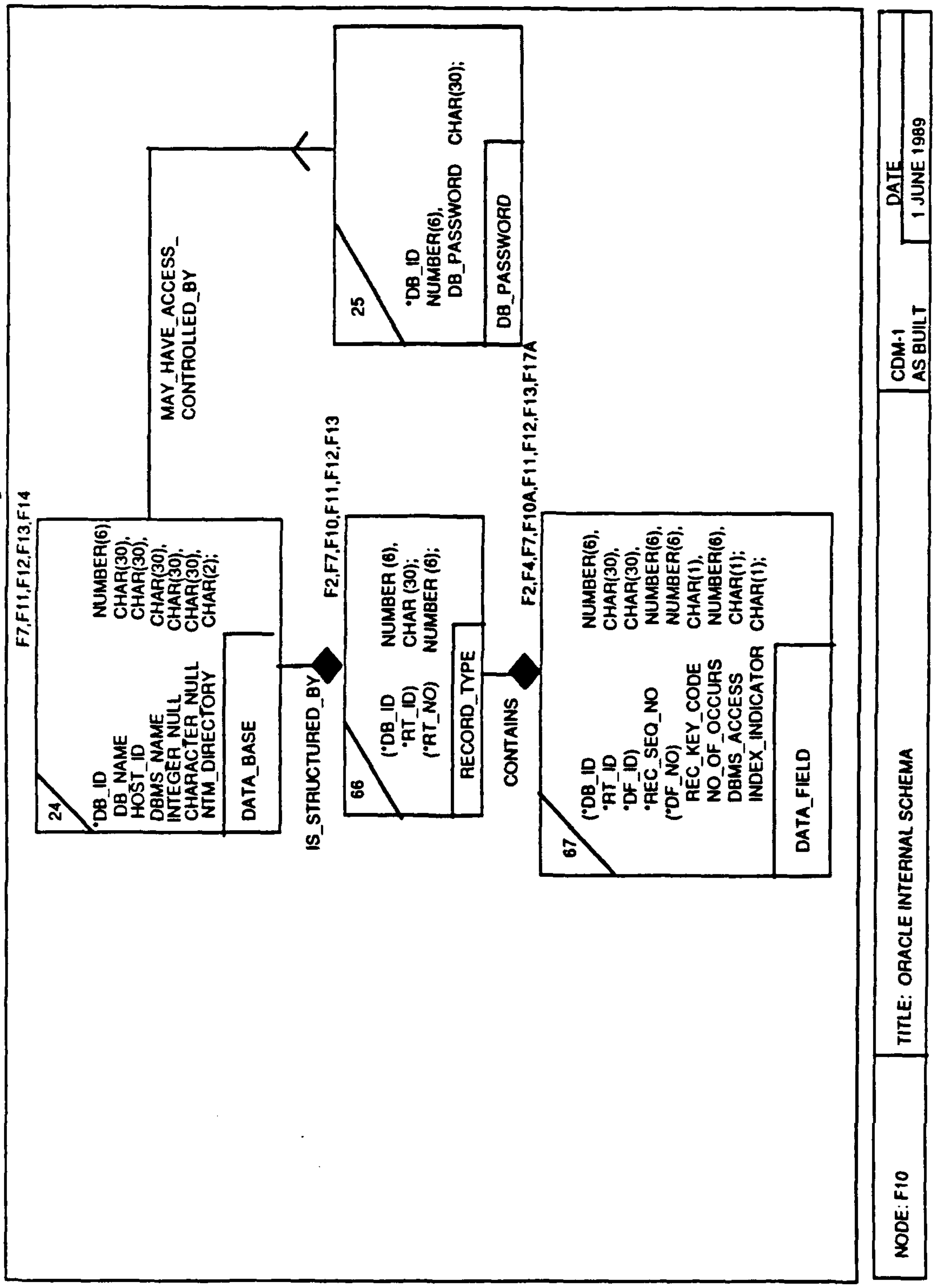

Figure 3-10. F10: ORACLE Internal Schema

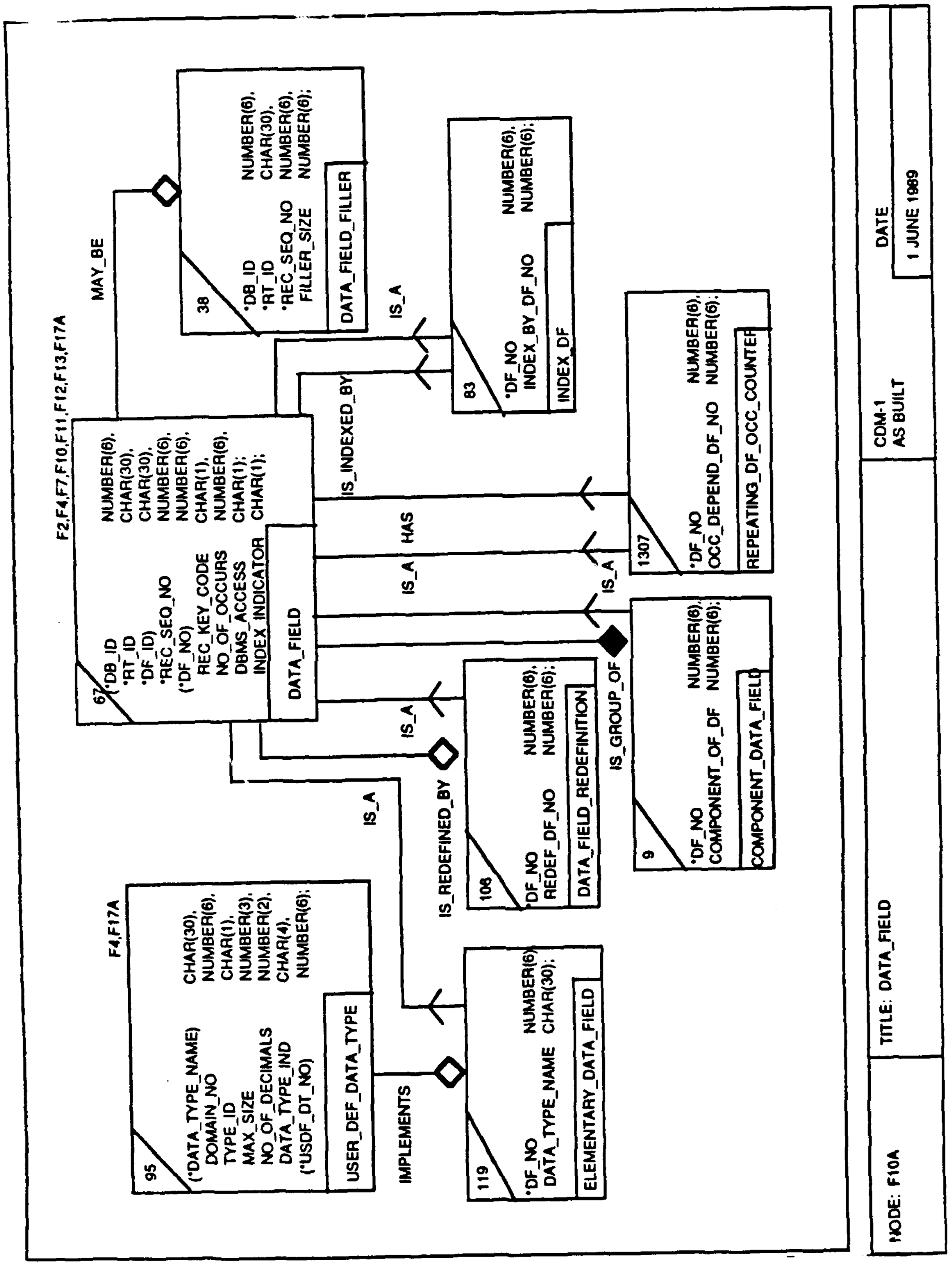

Figure 3-11. F11: Data Field

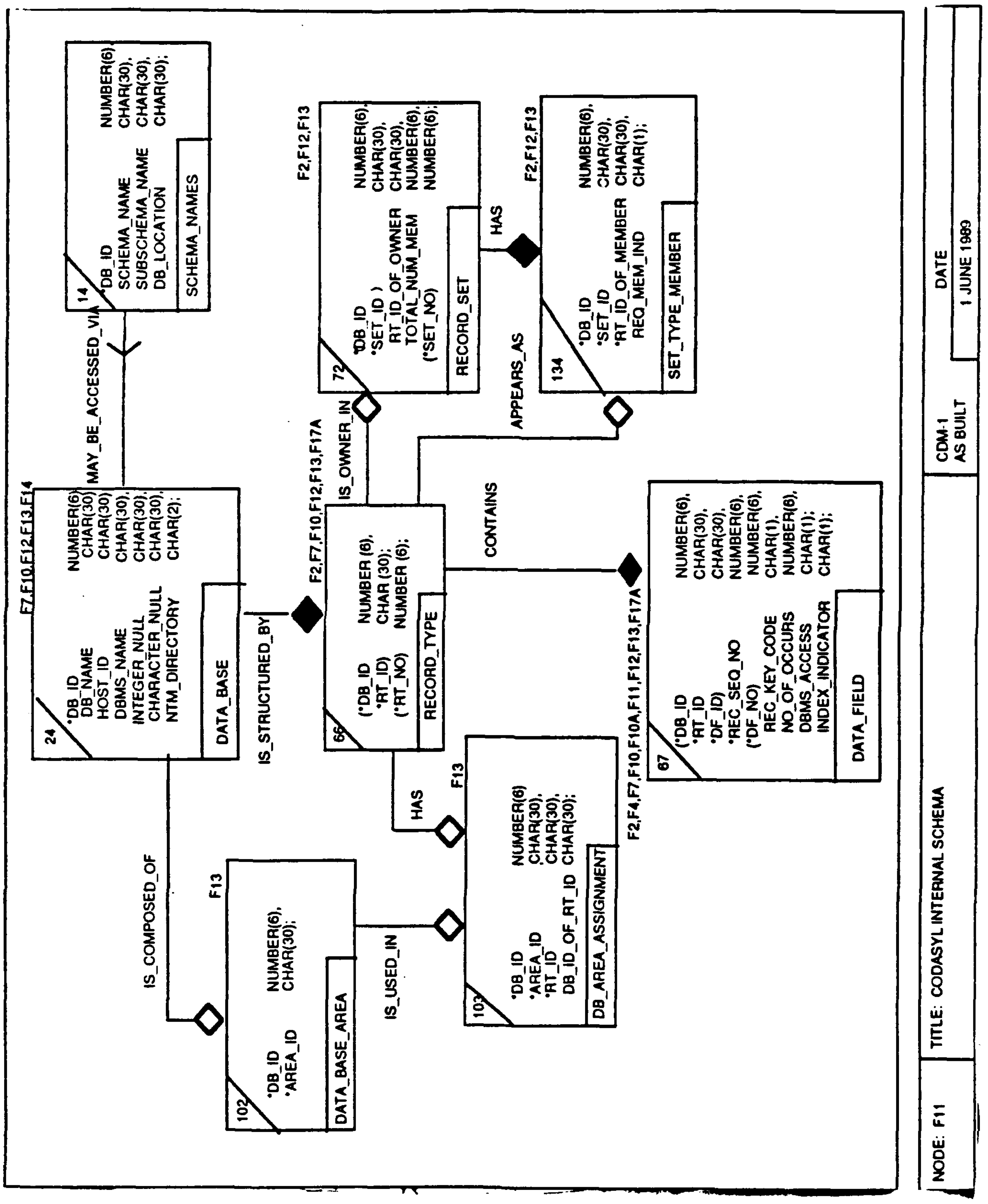

Figure 3-12. F12: CODASYL Internal Schemas

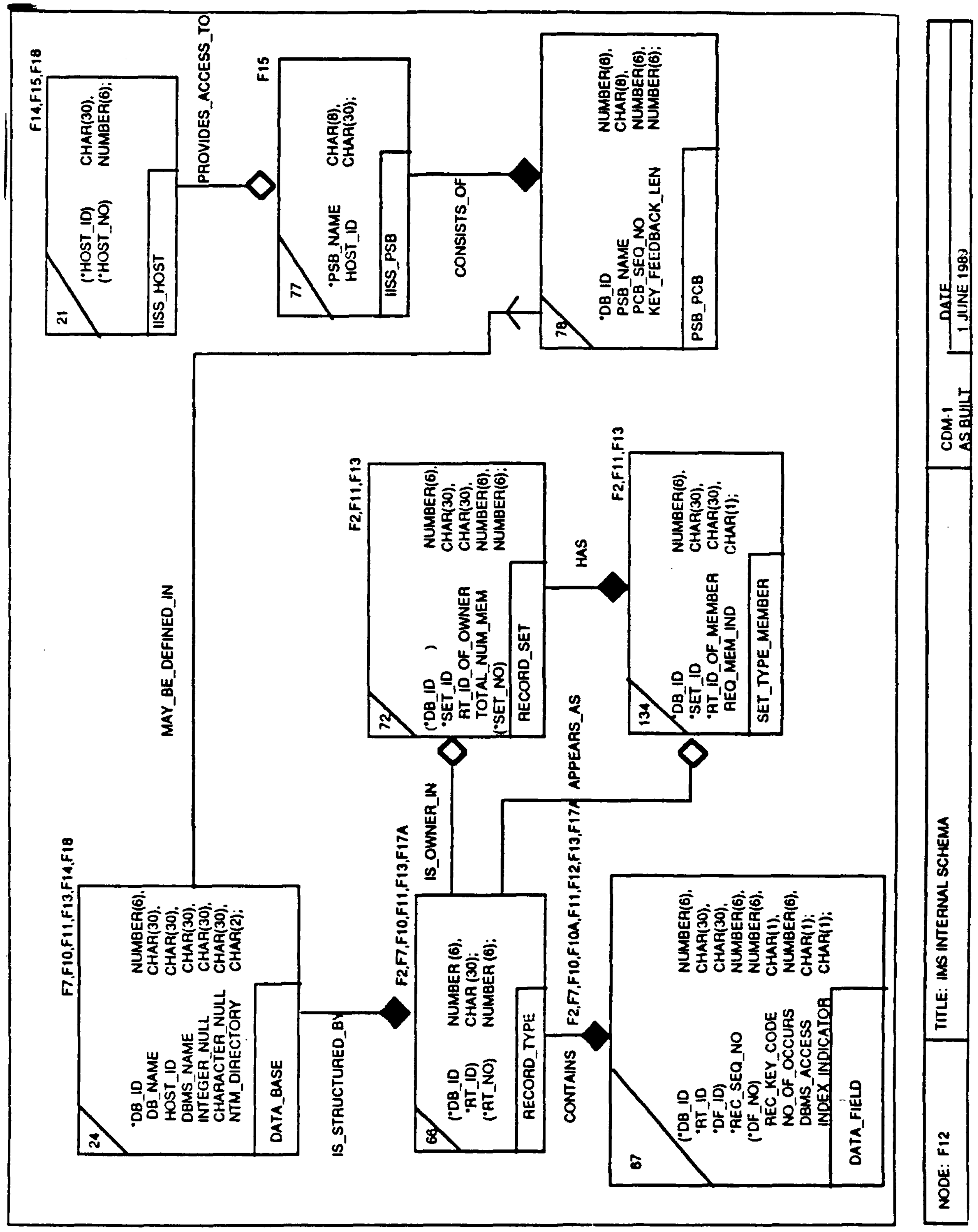

Figure 3-13. F13: IMS Internal Schema

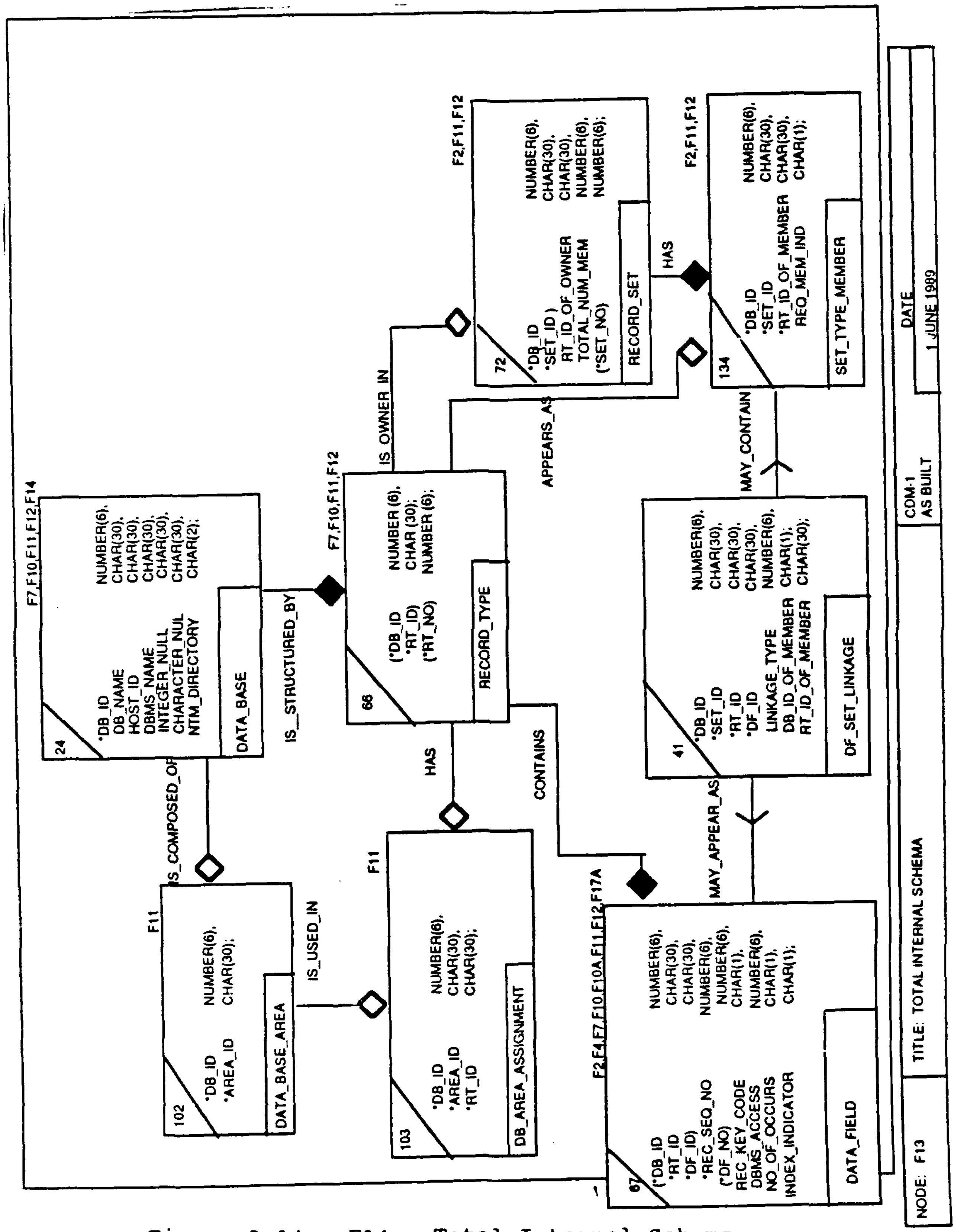

Figure 3-14. F14: Total Internal Schema

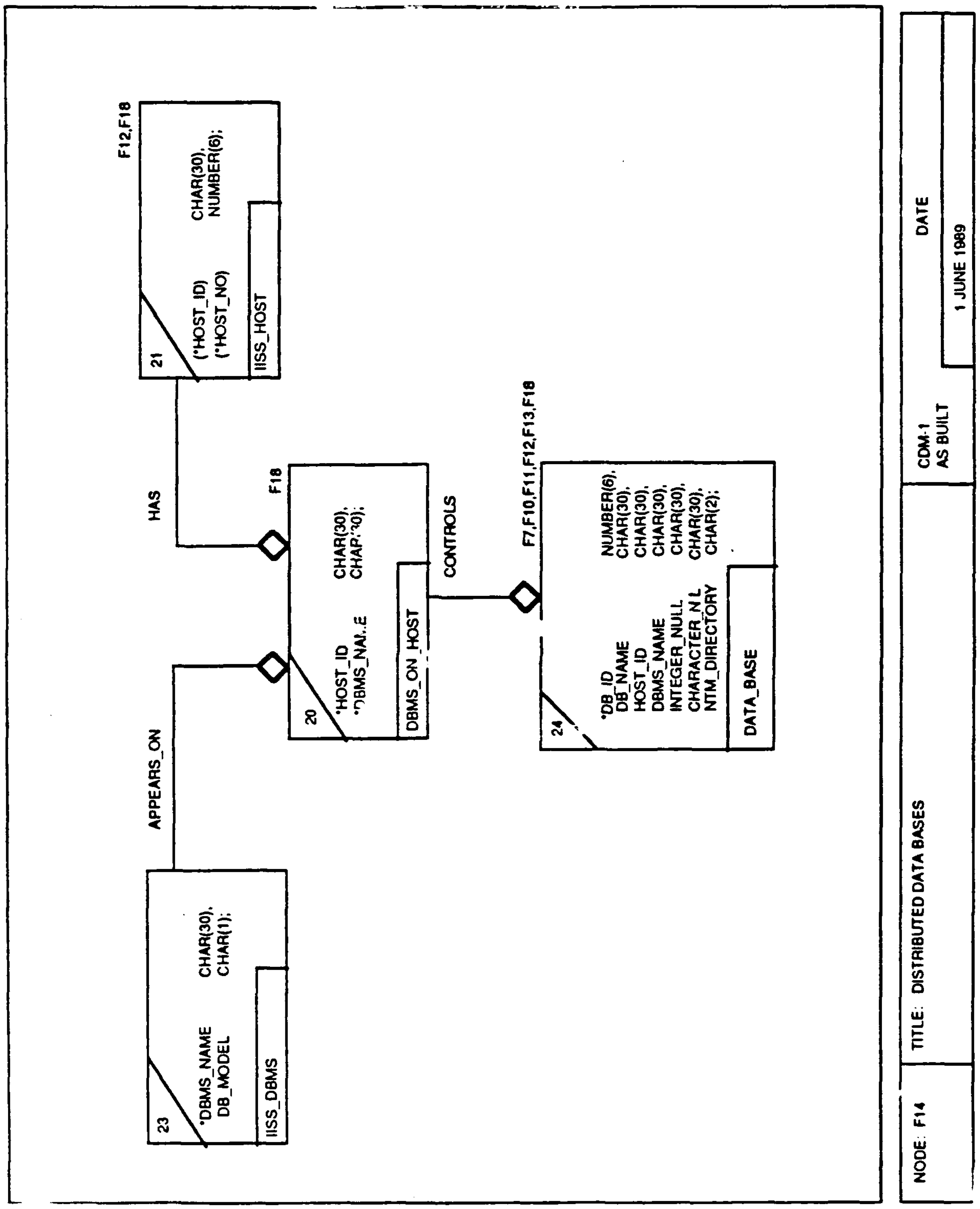

Figure 3-15. F15: Distributed Databases

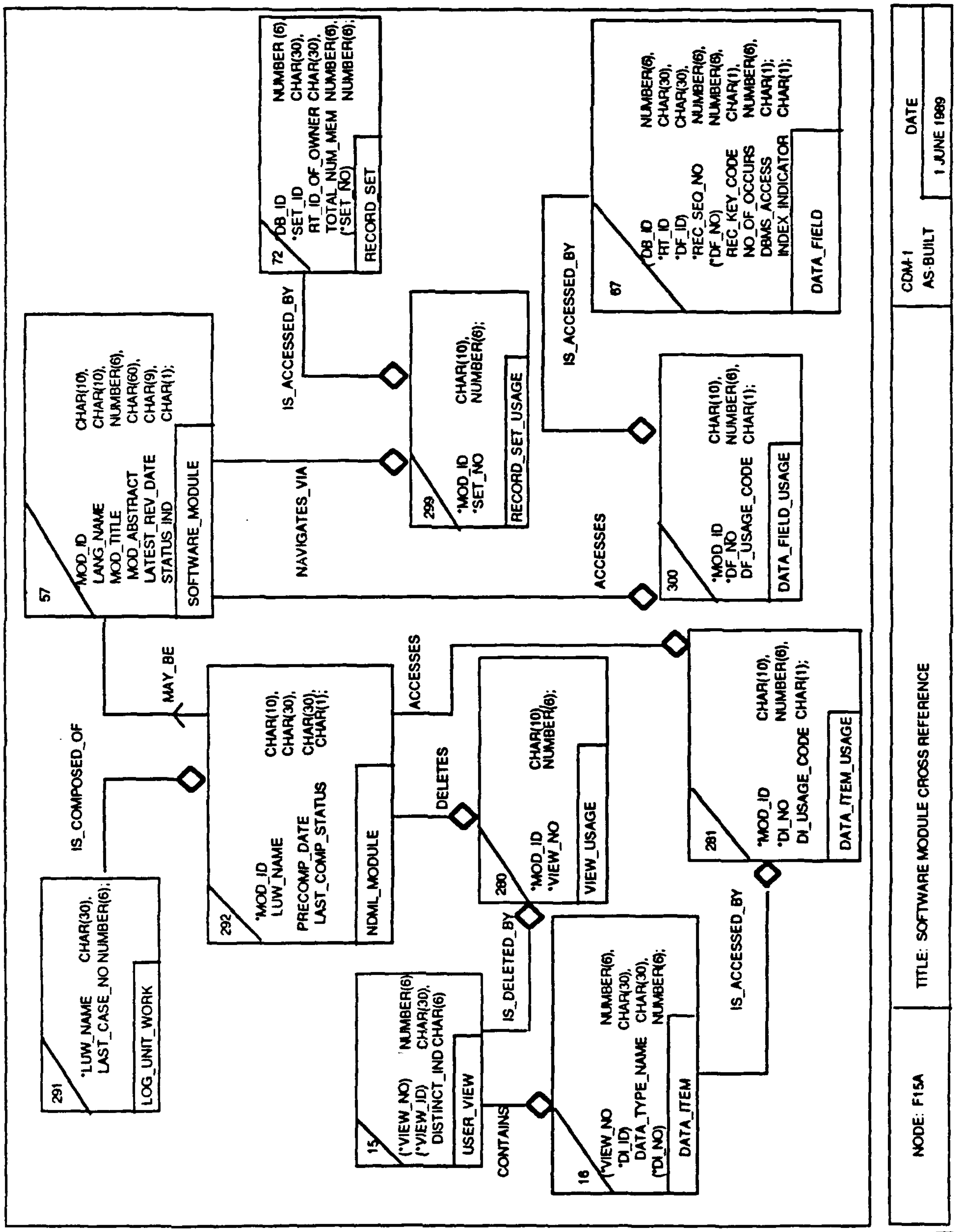

Figure 3-16. F16: Software Module Cross Reference

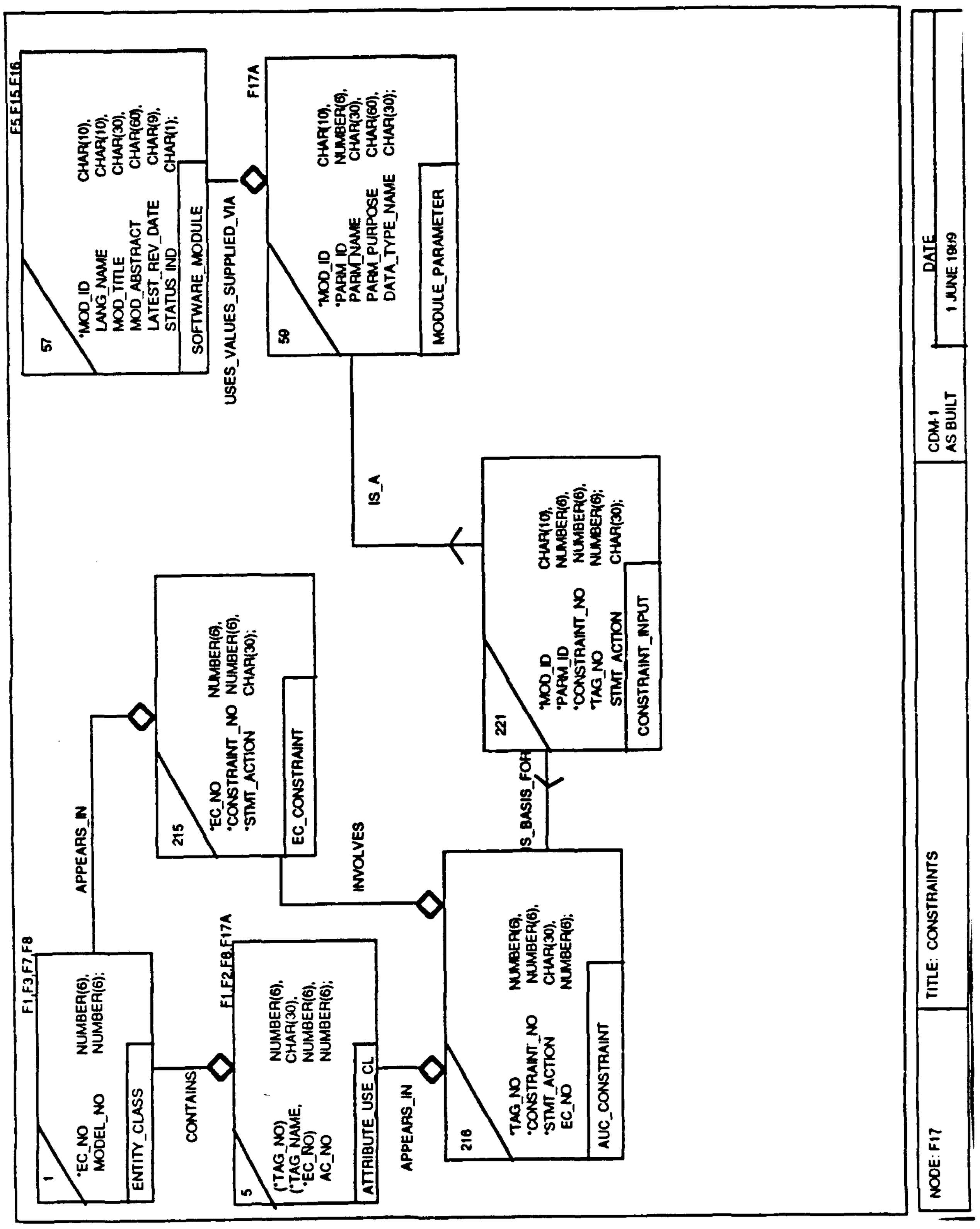

Figure 3-17. F17: Constraints

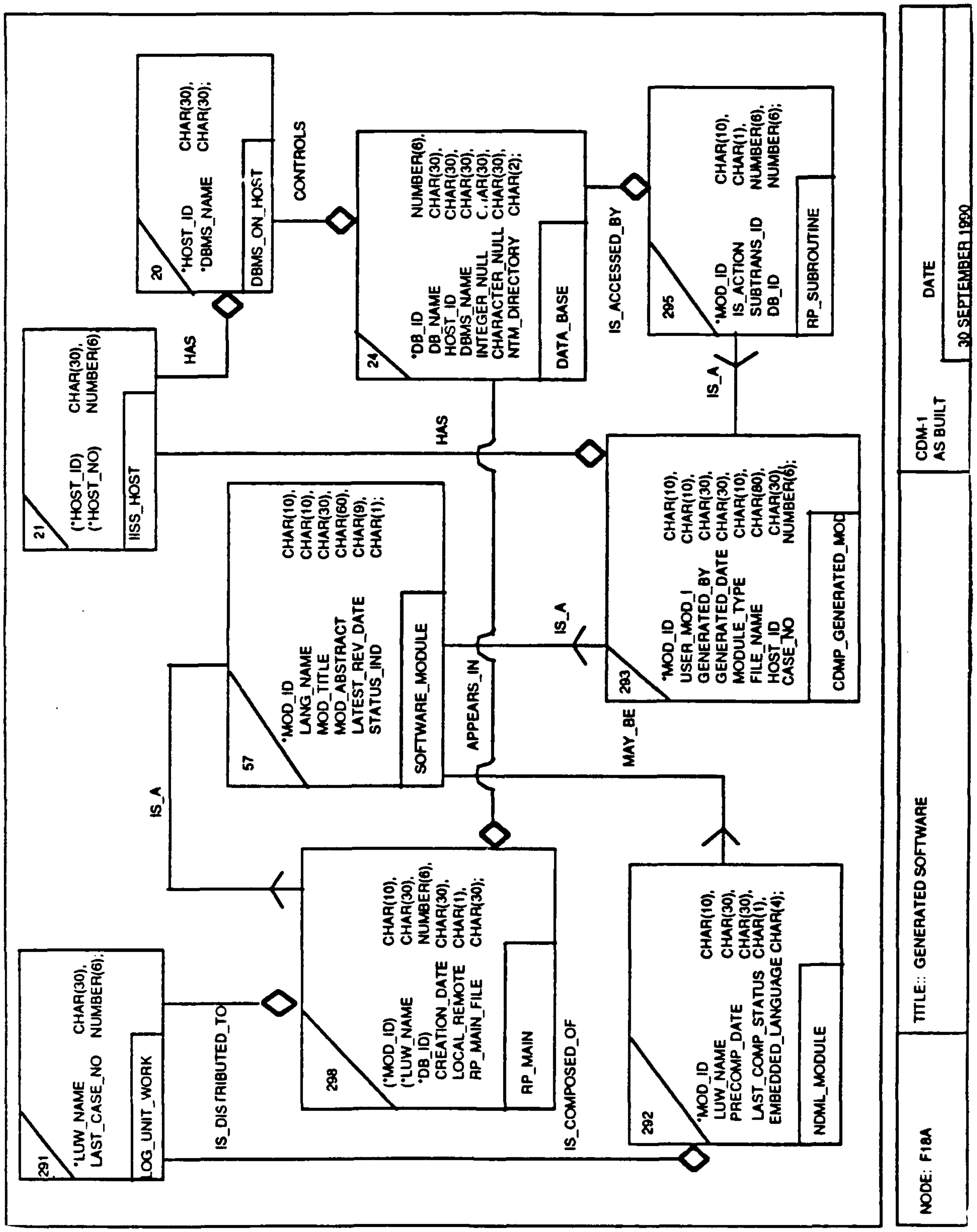

Figure 3-18. F18: Generated Software

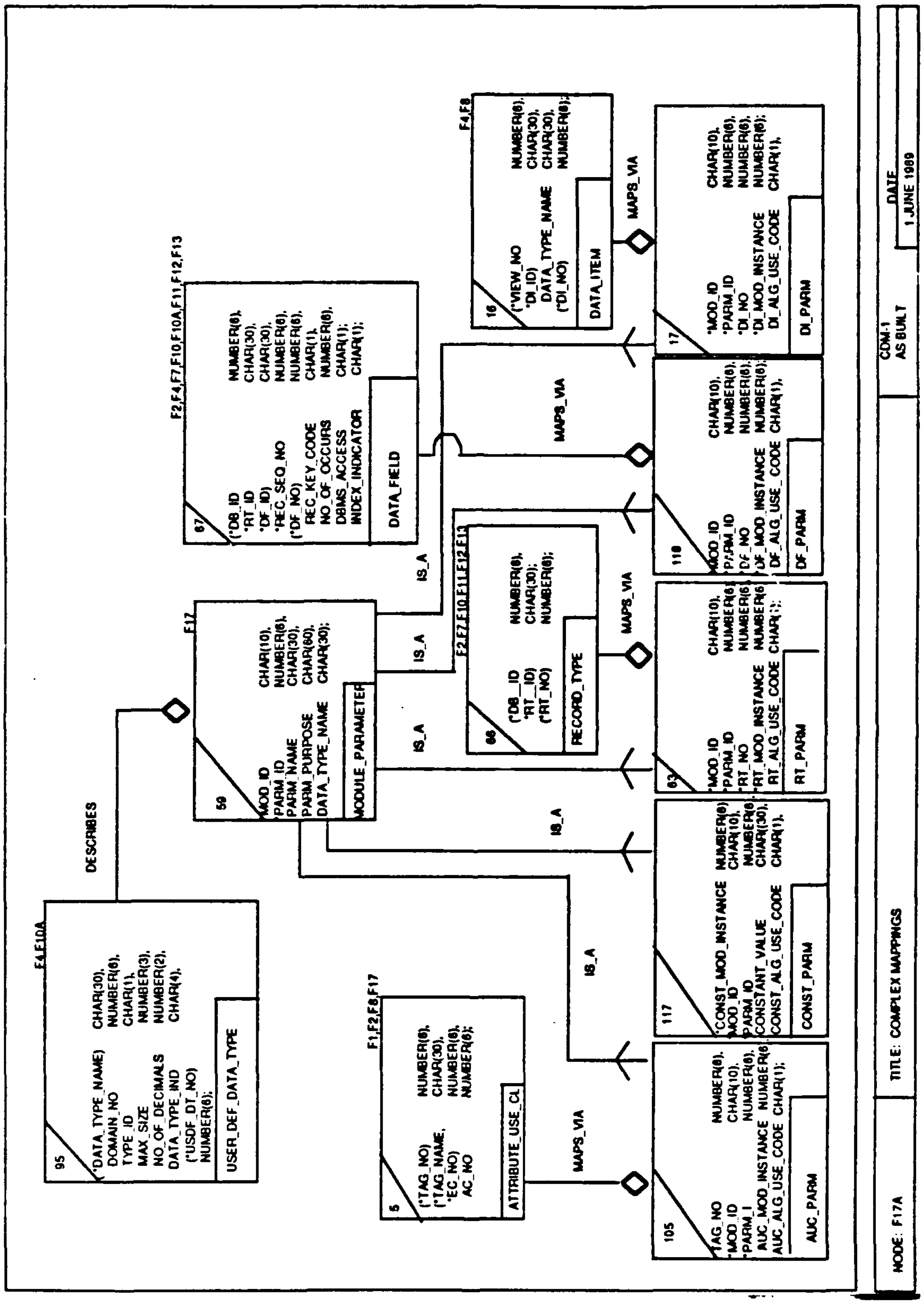

Figure 3-19. F19: Complex Mappings

SECTION 4

ENTITY CLASS DOCUMENTATION

4.1 CDM1 Entity Class Glossary

Note:
The entity classes in this glossary are in alphabetic sequence by entity class name.

AC_KEYWORD

The attribute class keyword provides the association between an attribute and a keyword descriptor.

ATTRIBUTE_CLASS

An attribute is a single unit of data about a real or abstract thing. An attribute consists of a name (e.g., employee hire date) and a value (e.g., 15 August 1980). An attribute class is a collection of all of the attributes which have the same meaning.

An attribute value may be:

A. Nondivisible (e.g., state name)

B. Divisible, i.e., a concatenation of two or more other attribute values (e.g., part number formed by concatenating drawing number and material code).

C. Computed from one or more other attribute values (e.g., age computed as current date minus birth date).

ATTRIBUTE_NAME

This entity provides the primary and alias identifiers of an attribute.

ATTRIBUTE_USE_CL

An attribute use class indicates that a particular attribute class is used in a particular entity class. The attribute class is either owned by that entity orinherited from another, related entity class.

AUC_IS_MAPPING

Indicates that an attribute use class corresponds to some portion of an internal schema.

AUC_PARM

An attribute use class that is used as a conceptual to internal or conceptual to external mapping parameter.

AUC_ST_MAPPING

Certain attribute use classes can be represented in a database by a group of record sets rather than by a data field. For example, Project Task Status might be represented by four PROJECT to TASK record sets called PENDING, IN-PROCESS, ON-HOLD, and COMPLETED. An attribute use class record set mapping indicates that a particular record set corresponds to a particular attribute use class value.

CATEGORY_MEMBER

CATEGORY_MEMBER contains information about a category relation's member entity.

CATEGORY_RELATION

CATEGORY_RELATION contains information which describes category relations,an extension of the IDEF methodology which allows subtype definitions based on a particular value of a discriminating attribute in the independent (generic) entity.

CDMP_GENERATED_MOD

Any software module (subroutine or main program), generated by the NDML Precompiler is a CDMP Generated Module. This definition includes the modified version of the user written (or generated) NDML module.

COMPLETE_RELATION

Also known as Inherited Key Class. A key class in the independent entity class of a relation that has migrated to appear in the dependent entity class of that relation class. The relation is known as complete since a key class has been migrated for it, thus making it specific.

COMPONENT_DATA_FIELD
A data field that is part of another data field, e.g., if data field A is made up of data fields B, C and D, each of these latter data fields is a component of A. A data field cannot be a component of more than one other data field.

CONSTRAINT_INPUT

A module parameter that is used to supply an attribute use class value to a generated constraint AP.

CONST_PARM

A static data value that is provided to a complex mapping algorithm, via a parameter, to indicate what the algorithm is to do.

DATA_BASE

A data base is a structurally interrelated collection of data whose interrelationships are identified to a DBMS for efficient and effective user access.

-For IMS:
DATA_BASE shall be used for each logical hierarchical view of IMS data. It must have a single root segment and be uniquely identified with a PCB (Program Communication Block).

-For TOTAL:
DATA_BASE will correspond to one Data Base Descriptor Module (DBMOD) within TOTAL.

DATA_BASE_AREA

A data base area is a subdivision of a Codasyl data base. This subdivision is a technique for improving the efficiency of accessing record type instances. When a data base is subdivided into data base areas, some or all of its record type are assigned to particular areas. Instances of these record types are stored only within the assigned areas. Then, these record type instances can be accessed by searching only the appropriate areas rather than the entire data base. This access method is only used when the record type instances cannot be located by calc keys or record sets.

-For TOTAL:
Each TOTAL data set will correspond to one DATA_BASE_AREA.

DATA_FIELD

A portion of a record type in which data values can be stored.

DATA_FIELD_FILLER

An occurrence of this entity class represents the fact that an area of a record (a data field, or contiguous groups of elementary data fields) is to be treated by the IISS as FILLER, or in other words, not as common data.

DATA_FIELD_REDEFINITION

A data field that occupies the same space in a record type as another data field. A record instance cannot contain values in both data fields. One instance can contain a value in one field while another contains a value in the other.

DATA_FIELD_USAGE

An occurrence of this entity will indicate that a particular data field is referenced by the named software module. The software module may be generated by the NDML Precompiler or previously exist as a user written, non-integrated application. The NDML Precompiler will only maintain entries for software modules it has generated.

DATA_ITEM

An attribute class as seen by a user in a user view, i.e., a kind of data (e.g., employee hire date), not a particular data value (e.g., 15 August 1980).

DATA_ITEM_USAGE

An occurrence of this entity will indicate that a particular data item has been referenced in an NDML statement found in an NDML module.

DATA_TYPE

The combination of a type of value (e.g. alphanumeric, signed numeric, etc.) and a type of storage (e.g. binary, packed, etc.).

DBMS_ON_HOST

A data base management system on host computer is a data base manager that operates on a specific host computer.

DB_AREA_ASSIGNMENT

A data base area assignment indicates that a particular record type is assigned to a particular data base area.

DB_PASSWORD

Many dbms's, not all, restrict usage by means of a password. This entity class will represent that password needed by generated AP's to access the particular data base.

DESCRIPTION_TYPE

This entity contains a word which identifies the type of text entered for the generic object description, i.e. DEFINITION, EXPANDED_NAME, POLICY, USAGE, etc.

DESC_TEXT

One fixed length portion of a generic object description.

DF_PARM

A data field that is used as a conceptual to internal mapping parameter.

DF_SET_LINKAGE

DF_USED_AS_SET_LINKAGE indicates that a particular data field is a symbolic calc key or a data base key that is used for a set linkage to the owner record occurrence.

-For TOTAL
DF_USED_AS_SET_LINKAGE will be used to record the TOTAL data element defined as the variable control key for the linkpath defined in the set from the variable data set to the master data set. This must be the element that was defined in the Data Base Descriptor Module as the variable control key for the linkpath.

DI_PARM

A data item that is used as a external to conceptual mapping parameter.

DOMAIN_CLASS

The set of rules about the values that are allowed for a data item, attribute class or data field. A domain is either an elementary domain or a group of two of more elementary domains, called component domains. Only elementary domains have been implemented.

DOMAIN_RANGE

A domain range is a series of consecutive values that represent all or part of an elementary domain;

DOMAIN_VALUE

A domain value is a single value within an elementary domain.

ECRTUD

If a record type corresponds to more than one entity class, i.e., if it has more than one entity to record type mapping,it is a relational union of those entity classes. Some instances of such a record type correspond to instances of one of the entity classes, others to those of another. For such a record type there must be a way to determine which record instances correspond to instances of each entity class. An entity class/record type union discriminator provides this by specifying that a given value in a given data field indicates that a given entity to record type mapping should be used.

EC_KEYWORD

This entity class provides the association between an entity and a keyword descriptor.

EC_RT_JOIN

A relational operation that combines two related entities as part of the design of a record type.

EC_RT_MAPPING

Indicates that an entity class corresponds to a record type, i.e., that they both have the same meaning and that the record type can be used to store instances of the entity. If a record type has more than one EC-RT mapping, some of its instances corresponding to instances of one entity class while others correspond to instances of another, i.e., the record type is the relational union of the entity classes. An example is a Replenishment Order record type that maps to both the Purchase Order and Manufacturing Order entity classes. Each record instance represents either a purchase order or a manufacturing order.

ELEMENTARY_DATA_FIELD

A data field that does not have any component data fields.

ENTITY_CLASS

An entity is a group of attributes about the same real or abstract thing that conform to the no-repeat and no-null rules and that can be uniquely identified by some of those attributes, singly or in combination. An entity class is a collection of all the entities that meet the following conditions:

1. Each entity in the collection contains attributes from the same attribute classes as every other entity in the collection.

2. Each entity in the collection is allowed to have the same relationships as every other entity in the collection.

ENTITY_NAME

This entity provides the primary and alias identifiers of an entity.

HORIZONTAL_PART

Indicates that the same record type is not used to store all instances of an entity class, i.e., that one is used to store some instances while another is used to store others. Each record type represents a fragment of the entity class. These fragments do not overlap, i.e., no entity class instance appears in more than one fragment. An entity class can be partitioned into any number of fragments, usually with each being in a different database or file, although that is not a requirement, some or all may be stored as different record types in the same database or file. A constraint statement defines each fragment, i.e., describes the conditions that must be met by each entity instance that is stored as a given record type. If an entity class is replicated, i.e., if each of its instances is stored in more than one database or file, each replication can be horizontally partitioned. For example, for the first replication the instances could be partitioned based on the values in one attribute use class, and for the second replication they could be partitioned based on the values in another.

IISS_DBMS

A data base management system is a set of computer programs that must be used to establish and maintain a data base.

IISS_HOST

A computer in the IISS.

IISS_KEYWORD

A keyword for a particular generic object. Keywords can be used to group many types of objects, across models around the same classifier. For example, relations, attributes and entities having something to do with the enterprises finance department or function could all be assigned to the same keyword FINANCE.

IISS_PSB

A Program Status Block (PSB) is used in IMS to group logical views of data bases into a single run time unit used for communicating with the IMS dbms.

INDEX_DF

An occurrence of this entity represents the fact that a data field is used as an index (in the COBOL sense) of another data field of the same record type.

INHERITED_ATT_USE

An inherited attribute use class is a key class member that has migrated from an independent entity class through a relation class to become an attribute use class in a dependent entity class.

KEY_CLASS

A key is a group of one or more of an entity's attributes that can be used to uniquely identify the entity within its entity class. An entity can have more than one key. A key class is a collection of the attribute classes whose member attributes comprise the keys for the entities in an entity class. An entity class has the same number of key classes as each of its member entities has keys. For example, if each entity has three keys, the entity class has three key classes.

KEY_CLASS_MEMBER

A key class member is an attribute use class that is part of a key class.

LINK_RELATION

An association between one entity in one entity calss and one in another. A relationship has a label that describes the association. For example, a customer named ABC corp. is associated with an order numbered 123 in a manner labeled 'placed'. A relation class is a collection of the identically labeled relationships between the members of the same two entity classes. Each relation class is either specific or nonspecific. In a specific relation class one entity class is independent while the other is dependent, i.e. entities in the first can exist without being associated with any in the second, but those in the second cannot exist without being associated with one in the first. One key class from the independent entity class migrates through each specific relation class to appear in the dependent entity class as inherited attribute classes.

In a nonspecific relation class, neither entity class is dependent on the other, i.e., entities in either entity class can exist without being associated with any in the other. For convenience, one entity class is arbitrarily called independent and the other is called dependent.

LOG_UNIT_WORK

A logical unit of work is a recoverable unit. A logical unit of work is a transaction. A recoverable unit has the following properties:

1. it obeys the rules of consistency

2. it either happens in its entirety or not at all

3. once it is committed, it cannot be undone.

A recoverable unit starts with the database in a consistent state. Its actions obey the system's consistency constraints. The initial consistent state is transformed into a new consistent state. A recoverable unit is atomic and durable. Either all of its actions are done and the unit is said to commit, or none of the effects of the unit survives and the unit is said to abort.

Each recoverable unit is defined to have exactly one of two outcomes: committed or aborted. It either completes successfully or it has no effect on the database state.

Once a recoverable unit commits, its effects can only be altered by running further recoverable units. It cannot be undone by the data base management system.

MODEL_CLASS

A model is a representation of the information requirements of all or part of an enterprise in terms of entity classes, relation classes, and attribute classes (more specifically IDEF-1).

MODULE_PARAMETER

A means of supplying values to a software module and of receiving results from a module.

NDML_MODULE

Any software module containing embedded NDML statement(s) that has been precompiled one time shall be known as an NDML module.

OWNED_ATTRIBUTE

An owned attribute class is a model attribute class that appears as an attribute use class in a model entity class and is not an inherited attribute use class.

PROJECT_DATA_FIELD

Indicates that an attribute use class corresponds to a data field, i.e. that they have the same meaning and that the data field can be used to store values for the attribute use class.

PROJECT_DATA_ITEM

Indicates that an attribute use class corresponds to a data item; i.e. that they have the same meaning and that the data item can be used to access values for the attribute use class.

PSB_PCB

This entity class maintains the cross reference of PCB's (Program Communication Blocks) used by IMS to define the logical database hierarchies and the PSB (Program Status Block) which maintains the run time link to the dbms.

RC_BASED_REC_SET

Indicates that a record set represents the same association as a relation class. If a record set has more than one member type, it may represent several relation classes, a different one for each member. Hence, this entity class is only indirectly dependent on record set (via record set member).

RC_KEYWORD

The relation class keyword entity provides the association between a relation class and a keyword descriptor.

RECORD_SET

A record set indicates that an instance of one record type 'owns' some number of instances of another record type.

Instances of the first are called owners; those of the second, members.

-For IMS:
SET_TYPE shall be used to describe each logical relationship between segments in the logical hierarchy. Sets are not named as such in IMS and the name (SET_ID) should be 'manufactured' from the relevant parent and child segment names.

-For TOTAL:
SET_TYPE shall be used to represent each linkage path of associated master and variable data sets within a TOTAL data base.

RECORD_SET_USAGE

An occurrence of this entity will indicate that a particular data base record set is referenced by the named software module. The set can be used for data base structure modification or simple traversal. The software module may be generated by the precompiler or previously exist as a user written, non-integrated application. The precompiler will only maintain entries for software modules it has generated.

RECORD_TYPE

A record is a group of data values that are stored together as a unit in a data base. A record type is the collection of all the records of the same kind; i.e., all the records that contain similar data values.

-For IMS:
RECORD_TYPE shall be used to record each logical sensitive segment named in the logical PCB of the data base being described.

-For TOTAL:
RECORD_TYPE shall be used to record each master data set and each variable data set in the data base being described.

REPEATING_DF_OCC_COUNTER

A data field whose data values indicate how many occurrences of a repeating data field actually contain values.

RP_MAIN

A generated software module that acts as a main routine or witch to all RP Subroutines of a single logical unit of work for a single data base. It provides all interfacing to or from the distributed request supervisor (DRS). It performs the functions of data base open, data base close, begin transaction, commit and rollback. It can be a subroutine callable directly from the DRS.

RP_SUBROUTINE

A generated software module used to carry out a single subtransaction with a single data base. A single NDML statment may yield many subtransactions because of the distributed nature of the data. A subtransaction may select, insert, delete, or modify a data base. A subtransaction may also be used to verify referential integrity constraints against a single data base by performing a select.

RT_PARM

A record type that is used as a conceptual to internal mapping parameter.

SCHEMA_NAMES

This entity provides the schema name and subschema name through which a CODASYL database must be accessed by IISS generated query and update processors.

SET_TYPE_MEMBER

A Set_type_member indicates that a particular record type is a member of a particular set type.

-For IMS:
SET_TYPE_MEMBER is used to describe the child segment of the IMS parent-child hierarchical relationships.

-For TOTAL:
SET_TYPE_MEMBER shall be used to describe the variable data set of the TOTAL linkpath being described by SET_TYPE.

SOFTWARE_MODULE

A set of computer instructions that are treated as a whole (i.e, stored, compiled, and executed together.).

USER_DEF_DATA_TYPE

A domain can have several different styles for representing its values. For example, the following are all styles of the same date:

12/31/84
31 DEC 84
December 31, 1984
84366

USER_VIEW

Formerly known as Surrogate Entity Class. A group of data items that a user wants to deal with as a group. It is similar to an entity class but does not necessarily meet all the conditions for being one, it can be thought of as an unnormalized entity class. A user view is often the result of combining several entity classes via relational join operations and selecting particular attribute use classes as data items via relational project operations.

VIEW_EC_XREF

The cross reference of what entity or entities make up a view. A view can be created from one or more entities. Each VIEW_NO can have at least one or more EC_NO's associated with it.

VIEW_QUALIFY_CRITERIA

A view may have qualify criteria (where clause) specified to limit its selection from certain entities that make it up. VIEW_QUALIFY_CRITERIA holds all information about the where clause of a view (if any). Each entry into it will be one singular piece of text. For example the where clause:

```
WHERE A.PART_NO = B.PART_NO AND

             A.DEPT_NO > 10
```

would have 7 entries for this view (VIEW_NO) :

```
A.PART_NO
     =
     B.PART_NO
     AND
     A.DEPT_NO
     >
     10
```

Includes: how many of these pieces of text are in this view, the number of conditions (expression-operator-expression) in the clause, the type of each condition, the type of each piece of text, and the actual text.

VIEW_QUAL_XREF

The entity to hold the cross reference between a view-entity combination and a tag that is used in the where clause of a CREATE VIEW command. If no where clause is specified when creating the view there will no data in this entity for that VIEW_NO.

VIEW_USAGE

An occurrence of this entity will indicate that a particular external schema view has been referenced as the object of the NDML DELETE statement in the given software module.

4.2 Owned and Inherited Attribute Classes

Entity Name	Owned Attributes	Inherited Attribute Role Name	Inherited From
AC_KEYWORD		AC_NO	ATTRIBUTE_CLASS
		KW_NO	IISS_KEYWORD
ATTRIBUTE_CLASS		DOMAIN_NO	DOMAIN_CLASS
		MODEL_NO	MODEL_CLASS
	AC_NO		
ATTRIBUTE_NAME		AC_NO	ATTRIBUTE_CLASS
	AC_NAME		
	AC_NAME_TYPE		
ATTRIBUTE_USE_CL		AC_NO	OWNED_ATTRIBUTE
		EC_NO	ENTITY_CLASS
	TAG_NAME		
	TAG_NO		
AUC_CONSTRAINT		CONSTRAINT_NO	EC_CONSTRAINT
		EC_NO	EC_CONSTRAINT
		STMT_ACTION	EC_CONSTRAINT
		TAG_NO	ATTRIBUTE_USE_CL
AUC_IS_MAPPING		EC_NO	EC_RT_MAPPING
		RT_NO	EC_RT_MAPPING
		TAG_NO	ATTRIBUTE_USE_CL
	MAP_CATEGORY		
	MAP_CLASS		
	MAP_TYPE		
	PREF_NO		
AUC_PARM		MOD_ID	MODULE_PARAMETER
		PARM_ID	MODULE_PARAMETER
		TAG_NO	ATTRIBUTE_USE_CL
	AUC_ALG_USE_CODE		
	AUC_MOD_INSTANCE		

Entity Name	Owned Attributes	Inherited Attribute Role Name	Inherited From
AUC_ST_MAPPING		DB_ID	RECORD_SET
		EC_NO	AUC_IS_MAPPING
		RT_NO	AUC_IS_MAPPING
		SET_ID	RECORD_SET
		TAG_NO	AUC_IS_MAPPING
	AUC_VALUE		
CATEGORY_MEMBER		DOMAIN_NO	DOMAIN_VALUE
		EC_NO	ENTITY_CLASS
		RC_NO	CATEGORY_RELATION
		SPECIFIC_VALUE	DOMAIN_VALUE
CATEGORY_RELATION		EC_NO	ENTITY_CLASS
		RC_NO	RELATION_CLASS
		TAG_NO	ATTRIBUTE_USE_CL
	CATEGORY_RC_NAME		
	CAT_TYPE_CODE		
CDMP_GENERATED_MOD		HOST_ID	IISS_HOST
		MOD_ID	SOFTWARE_MODULE
		USER_MOD_ID	NDML_MODULE
	CASE_NO		
	FILE_NAME		
	GENERATED_BY		
	GENERATED_DATE		
	MODULE_TYPE		
COMPARE_RESULTS	CAT_NAME		
	CDM_VERS		
	COMP_REASON		
	ITM_NAME		
	LEV_NAME		
	SCHEMA_ID		
COMPLETE_RELATION		KC_NO	KEY_CLASS
		RC_NO	RELATION_CLASS
COMPONENT_DATA_FIELD		COMPONENT_OF_DF	DATA_FIELD
		DF_NO	DATA_FIELD

Entity Name	Owned Attributes	Inherited Attribute Role Name	Inherited From
CONSTRAINT_INPUT		CONSTRAINT_NO	AUC_CONSTRAINT
		MOD_ID	MODULE_PARAMETER
		PARM_ID	MODULE_PARAMETER
		STMT_ACTION	AUC_CONSTRAINT
		TAG_NO	AUC_CONSTRAINT
CONST_PARM		MOD_ID	MODULE_PARAMETER
		PARM_ID	MODULE_PARAMETER
	CONSTANT_VALUE		
	CONST_ALG_USE_CODE		
	CONST_MOD_INSTANCE		
DATA_BASE		DBMS_NAME	DBMS_ON_HOST
		HOST_ID	DBMS_ON_HOST
	CHARACTER_NULL		
	DB_ID		
	DB_NAME		
	INTEGER_NULL		
	NTM_DIRECTORY		
DATA_BASE_AREA		DB_ID	DATA_BASE
	AREA_ID		
DATA_FIELD		DB_ID	RECORD_TYPE
		RT_ID	RECORD_TYPE
	DBMS_ACCESS		
	DF_ID		
	DF_NO		
	INDEX_INDICATOR		
	NO_OF_OCCURS		
	REC_KEY_CODE		
	REC_SEQ_NO		
DATA_FIELD_FILLER		DB_ID	DATA_FIELD
		REC_SEQ_NO	DATA_FIELD
		RT_ID	DATA_FIELD
	FILLER_SIZE		
DATA_FIELD_REDEFINI		DF_NO	DATA_FIELD
TION			

Entity Name	Owned Attributes	Inherited Attribute Role Name	Inherited From
DATA_FIELD_REDEFINITION		REDEF_DF_NO	DATA_FIELD
DATA_FIELD_USAGE		DF_NO	DATA_FIELD
MOD_ID	SOFTWARE_MODULE		DF_USAGE_CODE
DATA_ITEM		DATA_TYPE_NAME	USER_DEF_DATA_TYPE
		VIEW_NO	USER_VIEW
	DI_ID		
	DI_NO		
DATA_ITEM_USAGE		DI_NO	DATA_ITEM
		MOD_ID	NDML_MODULE
	DI_USAGE_CODE		
DATA_TYPE	TYPE_DESC		
	TYPE_ID		
DBMS_ON_HOST		DBMS_NAME	IISS_DBMS
		HOST_ID	IISS_HOST
DB_AREA_ASSIGNMENT		AREA_ID	DATA_BASE_AREA
		DB_ID	DATA_BASE_AREA
		DB_ID_OF_RT_ID	RECORD_TYPE
		RT_ID	RECORD_TYPE
DB_PASSWORD		DB_ID	DATA_BASE
	DB_PASSWORD		
DESCRIPT-ION_TYPE	DESC_TYPE		
	DESC_TYPE_ID		
DESC_TEXT		DESC_TYPE	DESCRIPTION_TYPE
	DESC_TEXT		
	LINE_NO		
	OBJECT_NO		
	OBJECT_TYPE		

Entity Name	Owned Attributes	Inherited Attribute Role Name	Inherited From
DF_PARM		DF_NO	DATA_FIELD
		MOD_ID	MODULE_PARAMETER
		PARM_ID	MODULE_PARAMETER
	DF_ALG_USE_CODE		
	DF_MOD_INSTANCE		
DF_SET_LINKAGE		DB_ID	DATA_FIELD
		DB_ID_OF_MEMBER	SET_TYPE_MEMBER
		DF_ID	DATA_FIELD
		RT_ID	DATA_FIELD
		RT_ID_OF_MEMBER	SET_TYPE_MEMBER
		SET_ID	SET_TYPE_MEMBER
	LINKAGE_TYPE		
DISTRIBUTED_RULES		EC_NO	ENTITY_CLASS
	DISTR_RETR_RULE		
	DISTR_UPDT_RULE		
DI_PARM		DI_NO	DATA_ITEM
DI_PARM		MOD_ID	MODULE_PARAMETER
		PARM_ID	MODULE_PARAMETER
	DI_ALG_USE_CODE		
	DI_MOD_INSTANCE		
DOMAIN_CLASS	DOMAIN_NAME		
	DOMAIN_NO		
DOMAIN_RANGE		DOMAIN_NO	DOMAIN_CLASS
	BEGIN_VALUE		
	END_VALUE		
DOMAIN_VALUE		DOMAIN_NO	DOMAIN_CLASS
	SPECIFIC_VALUE		
ECRTUD		DF_NO	DATA_FIELD
		EC_NO	EC_RT_MAPPING
		RT_NO	EC_RT_MAPPING
	COMPARISON_OP		
	UNION_VALUE		

Entity Name	Owned Attributes	Inherited Attribute Role Name	Inherited From
EC_CONSTRAINT		EC_NO	ENTITY_CLASS
	CONSTRAINT_NO		
	STMT_ACTION		
EC_KEYWORD		EC_NO	ENTITY_CLASS
		KW_NO	IISS_KEYWORD
EC_RT_JOIN		EC_NO	EC_RT_MAPPING
		RC_NO	COMPLETE_RELATION
		RT_NO	EC_RT_MAPPING
	JOIN_TYPE		
EC_RT_MAPPING		EC_NO	ENTITY_CLASS
		RT_NO	RECORD_TYPE
ELEMENTARY_DATA_FIE LD		DATA_TYPE_NAME	USER_DEF_DATA_TYPE
		DF_NO	DATA_FIELD
ENTITY_CLASS		MODEL_NO	MODEL_CLASS
	EC_NO		
ENTITY_NAME		EC_NO	ENTITY_CLASS
	EC_NAME		
	EC_NAME_TYPE		
HORIZONTAL_PART		EC_NO	ENTITY_CLASS
		RT_NO	RECORD_TYPE
	HP_NO		
IISS_DBMS	DBMS_NAME		
	DB_MODEL		
IISS_HOST	HOST_ID		
	HOST_NO		
IISS_KEYWORD	CDM_KEYWORD		
	KW_NO		
IISS_PSB		HOST_ID	IISS_HOST

Entity Name	Owned Attributes	Inherited Attribute Role Name	Inherited From
IISS_PSB	PSB_NAME		
INDEX_DF		DF_NO	DATA_FIELD
		INDEX_BY_DF_NO	DATA_FIELD
INHERITED_ATT_USE		KCM_TAG_NO	KEY_CLASS_MEMBER
		KC_NO	KEY_CLASS_MEMBER
		RC_NO	COMPLETE_RELATION
		TAG_NO	ATTRIBUTE_USE_CL
KEY_CLASS		KEY_CLASS_EC_NO	ENTITY_CLASS
	KC_NAME		
	KC_NO		
	KEY_TYPE		
KEY_CLASS_MEMBER		KC_NO	KEY_CLASS
		TAG_NO	ATTRIBUTE_USE_CL
LINK_RELATION		DEP_EC_NO	ENTITY_CLASS
		IND_EC_NO	ENTITY_CLASS
LINK_RELATION		RC_NO	RELATION_CLASS
	LINK_RC_NAME		
	MAX_NO_DEP_ENT		
	MIN_NO_DEP_ENT		
	NO_IND_ENT		
LOG_UNIT_WORK	LAST_CASE_NO		
	LUW_NAME		
MODEL_CLASS	DATE_CREATED		
	DATE_MODIFIED		
	MODEL_NAME		
	MODEL_NO		
	MODEL_STATUS		
MODULE_PARAMETER		DATA_TYPE_NAME	USER_DEF_DATA_TYPE
		MOD_ID	SOFTWARE_MODULE
	PARM_ID		
	PARM_NAME		
	PARM_PURPOSE		

Entity Name	Owned Attributes	Inherited Attribute Role Name	Inherited From
NDML_MODULE		LUW_NAME	LOG_UNIT_WORK
		MOD_ID	SOFTWARE_MODULE
	LAST_COMP_STAT		
	PRECOMP_DATE		
OWNED_ATTRIBUTE		AC_NO	ATTRIBUTE_CLASS
		EC_NO	ENTITY_CLASS
PROJECT_DATA_FIELD		DB_ID	DATA_FIELD
		DF_ID	DATA_FIELD
		EC_NO	AUC_IS_MAPPING
		RT_ID	DATA_FIELD
		RT_NO	AUC_IS_MAPPING
		TAG_NO	AUC_IS_MAPPING
PROJECT_DATA_ITEM		DI_NO	DATA_ITEM
		EC_NO	VIEW_EC_XREF
		TAG_NO	ATTRIBUTE_USE_CL
PROJECT_DATA_ITEM		VIEW_NO	VIEW_EC_XREF
	PRIM_SECONDARY		
PSB_PCB		DB_ID	DATA_BASE
		PSB_NAME	IISS_PSB
	KEY_FEEDBACK_LEN		
	PCB_SEQ_NO		
RC_BASED_REC_SET		DB_ID	SET_TYPE_MEMBER
		RC_NO	RELATION_CLASS
		RT_ID	SET_TYPE_MEMBER
		SET_ID	SET_TYPE_MEMBER
RC_KEYWORD		KW_NO	IISS_KEYWORD
		RC_NO	RELATION_CLASS
RECORD_SET		DB_ID	RECORD_TYPE
		RT_ID_OF_OWNER	RECORD_TYPE
	SET_ID		
	SET_NO		

Entity Name	Owned Attributes	Inherited Attribute Role Name	Inherited From
RECORD_SET	TOTAL_NUM_MEM		
RECORD_SET_USAGE		MOD_ID	SOFTWARE_MODULE
		SET_NO	RECORD_SET
RECORD_TYPE		DB_ID	DATA_BASE
	RT_ID		
	RT_NO		
RELATION_CLASS	RC_NO		
	REL_TYPE		
REPEATING_DF_OCC_COUNTER		DF_NO	DATA_FIELD
		OCC_DEPEND_DF_NO	DATA_FIELD
RP_MAIN		DB_ID	DATA_BASE
		LUW_NAME	LOG_UNIT_WORK
		MOD_ID	SOFTWARE_MODULE
RP_MAIN	CREATION_DATE		
	LOCAL_REMOTE		
	RP_MAIN_FILE		
RP_SUBROUTINE		DB_ID	DATA_BASE
		MOD_ID	CDMP_GENERATED_MOD
	IS_ACTION		
	SUBTRANS_ID		
RT_PARM		MOD_ID	MODULE_PARAMETER
		PARM_ID	MODULE_PARAMETER
		RT_NO	RECORD_TYPE
	RT_ALG_USE_CODE		
	RT_MOD_INSTANCE		
SCHEMA_NAMES		DB_ID	DATA_BASE
	DB_LOCATION		
	SCHEMA_NAME		
	SUBSCHEMA_NAME		

Entity Name	Owned Attributes	Inherited Attribute Role Name	Inherited From
SET_TYPE_MEMBER		DB_ID	RECORD_SET
		DB_ID_OF_MEMBER	RECORD_TYPE
		RT_ID_OF_MEMBER	RECORD_TYPE
		SET_ID	RECORD_SET
	REQ_MEM_IND		
SOFTWARE_MODULE	LANG_NAME		
	LATEST_REV_DATE		
	MOD_ABSTRACT		
	MOD_ID		
	MOD_TITLE		
	STATUS_IND		
USER_DEF_DATA_TYPE		DOMAIN_NO	DOMAIN_CLASS
		TYPE_ID	DATA_TYPE
	DATA_TYPE_IND		
	DATA_TYPE_NAME		
USER_DEF_DATA_TYPE	MAX_SIZE		
	NO_OF_DECIMALS		
	USDF_DT_NO		
USER_VIE	DISTINCT_IND		
	VIEW_ID		
	VIEW_NO		
VIEW_EC_XREF		EC_NO	ENTITY_CLASS
		VIEW_NO	USER_VIEW
VIEW_QUALIFY_CRITERIA		VIEW_NO	USER_VIEW
	QC_CONDITION_NO		
	QC_TEXT		
	QC_TEXT_NO		
	QC_TEXT_TYPE		

Entity Name	Owned Attributes	Inherited Attribute Role Name	Inherited From
VIEW_QUAL_XREF		EC_NO	VIEW_EC_XREF
		TAG_NO	ATTRIBUTE_USE_CL
		VIEW_NO	VIEW_EC_XREF
VIEW_USAGE		MOD_ID	NDML_MODULE
		VIEW_NO	USER_VIEW

SECTION 5

ATTRIBUTE CLASS DOCUMENTATION

5.1 CDM1 Attribute Class Glossary

Note:
The attribute classes in this glossary are in alphabetic sequence by attribute class name.

AC_NAME

The attribute class name is an identifier of an attribute class as assigned by the modeler(s).

AC_NAME_TYPE

The attribute class name type identifies whether the attribute name is the preferred name or an alias. Values are:

PRIMARY	-	for the preferred name, and
ALIAS	-	for all alias names.

AC_NO

Role name for Object Number, introduced in the Attribute Class entity class.

AREA_ID

A data base area identification is a code that is assigned to a data base area. A data base area can be uniquely identified by the combination of its data base identification and its data base area identification.

-For TOTAL:
The four character alpha-numeric name for the data set identifier.

AUC_ALG_USE_CODE

A code that indicates whether an attribute use class parameter used in a Complex Mapping Algorithm can be used on retrieval or update. Legal values are R for retrieval and U for update. NOTE: When a search parameter is used at runtime it will use the complex mapping algorithm defined for update (U) because the value must be converted to conceptual format. The data being compared against uses the retrieval algorithm (R) because the data to be tested is converted to conceptual format.

AUC_MOD_INSTANCE

A sequence number that specifies which instance of a complex mapping algorithm usage the attribute use class is to be used as a parameter of.

AUC_VALUE

The attribute use class value is a value for an attribute use class that is represented by a set type.

BEGIN_VALUE

The lowest value in a domain range. Lowest depends on the definition of the standard data type for the domain.

CASE_NO

The actual case number assigned to a software module generated by the NDML Precompiler. See the definition of LAST_CASE_NO for uses of case numbers.

CATEGORY_RC_NAME

CATEGORY_RC_NAME captures the name of the category relation. This must be unique to a particular generic entity.

CAT_TYPE_CODE

CAT_TYPE_CODE contains the value (complete or incomplete) reflecting the type of category relation. A complete category is one in which each instance of an independent (generic) entity has exactly one category member instance. An incomplete category is one in which each instance of the generic entity may have at most one member instance.

CDM_KEYWORD

A keyword is a word or phrase that has been designated as a means of locating a generic object or a number of similar generic objects.

COMPARISON_OP

A relational operator (<, >, <=, >=, =, !=) used in forming a logical condition for a union discriminator data field with the associated union discriminator value.

CONSTANT_VALUE

CS-IS or ES-CS Mapping Constant Value. A static data value that is provided to a CS-IS or ES-CS complex mapping algorithm as a parameter.

CONSTRAINT_NO

Role name for Object Number; introduced in the Constraint Statement entity Class.

CREATION_DATE

The date the need for an RP MAIN module was detected by the NDML precompiler. Note, this is not neccessarily the same as the date the RP MAIN was generated since they may occur at different points in time.

DATA_TYPE_NAME

A noun or noun phrase that briefly describes and uniquely identifies a data type.

DATE_CREATED

The date the need for an RP MAIN module was detected by the NDML Precompiler. Note, this is not necessarily the same as the date the RP MAIN was generated since they may occur at different ponits in time.

DBMS_ACCESS

DBMS Accessible Data Field Indicator. A code that indicates whether a data field in a data base is visible to or hidden from the DBMS, i.e. whether the data field can be explicitly included in a DBMS transaction.

DBMS_NAME

A DBMS name is the name by which a data base management system is commonly known. Some examples are IMS, IDMS, System 2000, Model 204, TOTAL.

DB_ID

A data base identification is a code that is assigned to uniquely identify a data base.

DB_LOCATION

This attribute contains the fully qualified VAX directory name in which the VAX-11 data base is stored. This is used in generating the B statement in VAX-11 CODASYL COBOL programs.

DB_MODEL

A data base model indicates which model a data base is based on, such as relational, hierarchical, or network.

DB_NAME

A data base name is a name or code by which a DBMS uniquely identifies a data base.

-For IMS:
The 8 character name of the PCB defining the logical data base hierarchy shall be used.

-For TOTAL:
The six character alpha-numeric Data Base Descriptor Module (DBMOD) name.

DB_PASSWORD

A data base password is a code that the CDMP must supply when logging on to a DBMS to use a data base. The DBMS verifies the password before accepting any other messages from the CDMP.

DESC_TEXT

A fixed length portion of a generic object description.

DESC_TYPE

The description type is a word which identifies the type of text entered for the generic object description, i.e., DEFINITION, EXPANDED_NAME, POLICY, USAGE, etc.

DF_ID

A data field identification is a code that is assigned to a data field. A data field can be uniquely identified by the combination of its data base identification, its record type identification, and its data field identification.

-For IMS:
This shall be the 8 character IMS data field name. If the data field is not known to IMS, any legal COBOL name will do.

-For TOTAL:
The eight character name for a TOTAL data item with the following format:
mmmmxxxx

vvvvxxxx

where:	mmmm	- master data set name
	vvvv	- variable data set name
	xxxx	- four character entry as defined in the Data Base Descriptor Module.

DF_NO

Data Field Object Number. Role name for Object Number; introduced in the Data Field entity class.

DF_USAGE_CODE

This code identifies how a data field is used in a particular software module. Code values may be:

I-	data field is to be inserted
M-	data field is to be modified
S-	data field is to be selected for retrieval
Q-	data field is to be qualified or searched on
D-	data field is in a record or table which is to be deleted.

DISTINCT_IND

Distinct indicator will tell whether the data items being selected in the view are to be selected as:

Y - distinct or
N - not distinct

Since the distinct indicator applies to all the data items this attribute is owned at the view level in USER_VIEW.

DISTR_RETR_RULE

This attribute will indicate whether the CDMA wants the NDML Precompiler to generate retrieval subtransactions for secondary mappings of this entity in the event that the primary copy is on a different host than the one the NDML aplication is to execute on and an active mapping exists to a secondary copy on the target host of the NDML application. In the event that the CDMA does not specify, the default shall be DISALLOW.

DISTR_UPDT_RULE

This attribute will indicate whether the CDMA wants the NDML Precompiler to generate update subtransactions for all distributed mappings (non-primary) of this entity. in the event that the CDMA does not specify, the default shall be DISALLOW.

DI_ID

A data item identification is a code that is assigned to a data item. A data item can be uniquely identified by the combination of its surrogate entity class identification and its data item identification.

DI_NO

Data Item Object Number. Role name for Object Number, introduced in the Data Item entity class.

DI_USAGE_CODE

This code identifies how the data item is used in a particular NDML module. Code values may be:

I-	data item is to be inserted
M-	data item is to be modified
S-	data item is to be selected
Q-	data item is to be qualified on
D-	data item is in a view which is the object of a delete.

DOMAIN_NAME

The domain name attribute is the identifier of a domain.

DOMAIN_NO

The domain number is the numeric identifier of a domain.

EC_NAME

The entity class name attribute is any name assigned to an entity class by the modeler(s).

EC_NAME_TYPE

The entity class name type identifies whether the entity name is the preferred name for the entity or an alias. Values are:

PRIMARY	-	for the preferred name, and
ALIAS	-	for all alias names.

EC_NO

Role name for Object Number, introduced in the Entity Class entity class.

END_VALUE

Ending Value. The highest value in a domain range. Highest depends on the definition of the standard data type for the domain.

FILLER_SIZE

The size in characters of a data field to be defined as a filler. Value must be greater than zero.

GENERATED_BY

This attribute identifies the subcomponent of the NDML Precompiler which generated a software module.

HOST_ID

A host computer identification uniquely identifies a host computer.

HP_NO

A number that distinguishes one horizontal partition from any others for an entity class.

INDEX_INDICATOR

A code that indicates whether a data field is the index of another data field in the same record. Values may be Y for yes, N for no and G meaning a generated index which occupies no real data storage in the record (Y implies that the index actually occupies storage in the record and must have an elementary data field).

IS_ACTION

This attribute indicates the purpose of a request processor subroutine. The code values are:

I -	insert
M -	modify
D -	delete
S -	select
1 -	type 1 referential integrity test
2 -	type 2 referential integrity test
K -	type K referential integrity test.

JOIN_TYPE

A code to indicate how to join two entities in creating a record. The current implementation will only record and support outer join.

KC_NAME

The key class name is the name assigned to the key class by the user.

KC_NO

An identification code that distinguishes a key class from the others in the same model.

KEY_FEEDBACK_LEN

KEY_FEEDBACK_LEN is used in IMS PCB's to define the maximum size of the concatenated keys from each segment in the hierarchy from root to the bottom child segment. Since there may be many terminating child segments, KEY_FEEDBACK_LEN is the maximum. This parameter is available in the PCB.

KEY_TYPE

KEY_TYPE contains information describing whether the key class is primary or alternate.

LANG_NAME

A software language name is the name of the software language in which a software module is written. Some examples are COBOL, FORTRAN, and PASCAL.

LAST_CASE_NO

The last case number assigned to NDML conceptual subtransactions of a logical unit of work. A case may be an insert, modify, a delete, a select or any of the three types of referential integrity tests. Note, a case may be implemented with many subtransactions.

LAST_COMP_STAT

This attribute is used to record the last compile status of an ndml module. When the entity is stored, the status will be (N) for not yet compiled. After a successful precompiliation of the module, the status will be (S) and whenever a precompiliation for a module fails, the status will be (F). When request processor main programs are generated for a logical unit of work, no ndml modules for that logical unit of work may have a status of (N) or (F).

LATEST_REV_DATE

A latest revision date is the date when a software module was last revised or when it was first implemented if it has never been revised.

LINKAGE_TYPE

A LINKAGE_TYPE is a code that indicates whether the data field that is used for a set linkage is a symbolic calc key or an actual data base key. The values for this attribute are as follows:

S - symbolic calc key
K - data base key

-For TOTAL: This shall be a S indicating symbolic calc key.

LINK_RC_NAME

The relation class name is a verb phrase (i.e., a verb and its adverbs and prepositions) that briefly describes a module relation class and that distinguishes it form any others for the same two model entity classes.

LOCAL_REMOTE

A flag indicating whether the RP MAIN is a request processor which is subroutine called directly by the DRS (local) or as a stand alone main program accessed with NTM services (remote). Only one RP of the set needed for a logical unit of work can be local. L is used to indicate local and R for remote.

LUW_NAME

A user generated identified for a logical unit of work.

MAP_CATEGORY

This attribute shall be used to indicate that the conceptual to internal mapping for an attribute use clas is or is not to be used by the NDML precompiler when looking for valid versions or storage locations for the attribute use class.

MAP_CLASS

This attribute shall be used to distinguish among the various types of redundant data (as defined for the attributes domain) for the particular mapping of an attribute use class.

MAP_TYPE

A code that indicates the type of mapping defined for an attribute use class. Values may be:

FIELD	- mapping to a data field
SET	- mapping to record sets
COMPLEX	- mapping through use of a complex mapping algorithm.

MAX_NO_DEP_ENT

The greatest number of entities that are allowed to be dependent in instance of a relation class. Most relation classes have no exact maximum. For these, 99 is used to indicate that any number of entities may be dependent.

MAX_SIZE

The maximum size is the greatest number of characters (letters, numerals, punctuation, etc.) that can be represented by a data type, excluding any decimals.

MIN_NO_DEP_ENT

A minimum number of dependent entities indicates the least number of entities that are allowed to be dependent in a relation class.

MODEL_NAME

The model name attribute is the name assigned to the model by the user.

MODEL_NO

Role name for Object Number, introduced in the Model entity class.

MODEL_STATUS

This attribute is a flag used to indicate the status of the model as either checked or unchecked. It can only be set to checked by the NDDL CHECK MODEL command which examines a model for IDEF-1 completeness rules.

MODULE_TYPE

This attribute identifies the particular type of generated module. The choices are:

RP-SUB	- request processor subroutine
CS-ES	- conceptual to external transformer
CS-CS	- conceputal selector module
USER-MOD	- modified user software module.

MOD_ABSTRACT

A software module abstract is an explanation of what a software module does, how it does it, and how to use it.

MOD_ID

A software module identification is a code that is assigned to uniquely identify a software module.

MOD_TITLE

A software module title is a brief, descriptive name that is given to a software module.

NO_IND_ENT

The greatest number of entities that are allowed to be independent in any instance of a relation class. In a specific relation class this maximum is always one (1). In a non-specific relation class this may be zero or a number greater than one. Most non-specific relation classes have no exact maximum. For these, 99 is used to indicate that any number of entities may participate.

NO_OF_DECIMALS

The number of decimals is the greatest number of numerals that can be represented in the decimal portion of a data type.

NTM_DIRECTORY

This attribute contains the two character code for a directory in the NTM will find the executable image for generated request processors (RP). This allows the user to dictate a separate storage area for RP's associated with each data base.

OBJECT_NO

The number that uniquely identifies an instance of an object type, i.e. the OBJECT_NO of the object type DATA_FIELD is specific DF_NO.

OBJECT_TYPE

The following list of entities of the CDM shall be treated as OBJECTs, i.e. things that can be described:

MODEL, ENTITY, ATTRIBUTE, KEY CLASS, RELATION, TAG (ATTRIBUTE USE CLASS), DOMAIN, KEYWORD, USER VIEW, DATA BASE, RECORD SET, USER DEFINED DATA TYPE, DATA ITEM, DATA FIELD, RECORD TYPE, HOST.

PARM_ID

The ordinal position of a module parameter within the list of parameters for a software module.

PARM_NAME

A code that distinguishes a module parameter from the others for the same software module. The actual name used in the program may be used.

PARM_PURPOSE

A parameter purpose is an explanation of the role that a software module parameter has in relation to the function of the software module.

PCB_SEQ_NO

PSB_SEQ_NO indicates the ordinal position in the list of PCB's making up a PSB of a single PCB being referenced in the PCB_PSB cross reference.

PRECOMP_DATE

The latest date that the NDML module has been successfully precompiled.

PREF_NO

A number that indicates the relative acceptability of the values stored in one data field to those stored in another, both of which correspond to the same attribute use class. Preference numbers will be used by the CDMP to determine which data field to retrieve values from when multiple choices are available.

PRIM_SECONDARY

This attribute identifies a data item to attribute use class mapping as primary or secondary.

PSB_NAME

This attribute is the name of the IMS Program Status Block (PSB), a named area of memory used by the IMS database management system in communcicating with application processes.

QC_CONDITION_NO

A sequential number associated with a condition in the qualify criteria (where clause) of a CREATE VIEW command. Many pieces of text(QC_TEXT_NO) make up a condition. QC_TEXT_NO gives all the line numbers of text that make up each condition. See description type example for more information.

QC_COND_TYPE

The type of condition that is part of a qualify criteria (where clause) of a CREATE VIEW command. A condition can be either a join condition (column-operator-column) or a qualify condition (column-operator-value):

2 = Qualify condition

3 = Join condition

QC_TEXT

The actual piece of text specified in the where clause of a CREATE VIEW command. See definition of attribute QC_TEXT_TYPE for the types of pieces of text.

QC_TEXT_NO

A sequential number associated with each piece of text in a qualify criteria (where clause) of a CREATE VIEW command. A piece of text can be a comparison operator (=,<,>,>=,<=,!=,U=,NN,NL), a tag, a logical operator (AND,OR,(,),NOT), a literal constant, or a numeric constant. It is similar to the line number of the where clause.

QC_TEXT_TYPE

The type of piece of text that is part of a qualify criteria (where clause) of a CREATE VIEW command. A piece of text can be:

T = Tag
O = Comparison operator
    (=,<,>,<=,>=,!=,U=,NN,NL)
L = Logical opperator
    (AND,OR,(,),NOT)
C = Literal constant
V = Numeric constant

[illegible]C_NO

Role name for Object Number, introduced in the Relation Class entity class.

REC_KEY_CODE

A code that indicates whether the values in a data field can be used to locate instances of a record type and, if they can, whether they must be unique.

REC_SEQ_NO

The ordinal position of a data field within a record type.

REL_TYPE

This attribute distinguishes between link and category relations.

REQ_MEM_IND

A required membership indicator is a code that indicates whether instances of the member record type in a record set must be related to instances of the owner record type in order to exist. If they must be, the DBMS automatically deletes the member instances when an owner instance is deleted and does not permit a new member instance to be created without it being related to an owner instance. If a member instance does not have to be related to an owner instance in order to exist, the DBMS does not automatically delete member instances and does permit new member instances to be created without being related to owner instances. (Use 'R' for required, 'O' for optional)

-For IMS:
This shall be 'R' indicating required membership.

-For TOTAL:
This shall be 'R' indicating required membership.

RP_MAIN_FILE

The name of the file, determined at the time the RP need is detected, on which the RP MAIN should be generated. Note, this is not the name the RP will ultimately reside on after it is transferred to its host for compilation and execution.

RT_ID

A record type identification is a code that is assigned to a record type. A record type can be uniquely identified by the combination of its data base identification and its record type identification.

-For IMS:
This shall be the 8 character segment name.

-For TOTAL:
The four character alpha-numeric name for the data set identifier.

SCHEMA_NAME

The schema_name attribute is the name of the schema given a CODASYL data base.

SET_ID

A record set identification is a code that is assigned to a record set. A record set can be uniquely identified by the combination of its data base identification and its record set identification.

-For IMS:
This name shall be manufactured by combining the parent segment and child segment names and separating them with a dash.

-For TOTAL:
This name shall be the linkpath name as defined in the TOTAL Data Base Descriptor Module for a linkage from a master data set based on the record control key. The SET_ID is eight characters with the following format:

```
mmmmLKxx
where:    mmmm - master data set name
          xx   - two character linkage code as defined in
                 the Data Base
          Descriptor Module.
```

SET_NO

Role name for Object Number, introduced in the Record Set entity class.

SPECIFIC_VALUE

A value of an elementary domain.

STATUS_IND

A status indicator indicates if a software module identification is:

F	Available to be used
G	Is a generated module
B	Busy
C	Complex mapping algorithm

STMT_ACTION

An NDML action (SELECT, INSERT, MODIFY or DELETE) that is specified in the ON clause of a constraint statement.

SUBSCHEMA_NAME

The subschema name attribute is the name of the subschema given a CODASYL data base through which IISS access is controlled.

SUBTRANS_ID

This attribute serves to distinguish the many subtransactions, at the internal level, that may be needed to process a single conceptual transaction. The identifiers are assigned sequentially starting at 1 for each subtransaction of a conceptual transaction. Note, NDML COMMIT, ROLLBACK and BEGIN statements do not generate actual subtransactions since they are handled by the RP MAIN and not the RP SUBROUTINE.

TAG_NAME

A tag name is the identifier of an attribute use class. A tag uniquely identifies an attribute within an entity because an attribute may be used many times in the same or different entities. Also known as role name.

TAG_NO

A tag number is an identification code composed of the letter T followed by a number. It is assigned to uniquely identify an attribute use class.

TOTAL_NUM_MEM

A total number of members is a number that indicates how many record types are members in a database record set.

TYPE_DESC

A data type description is a complete, concise explanation of what a data type represents.

TYPE_ID

A data type identification is a code that is assigned to uniquely identify a data type.

-For IMS:
Types S, N, C and P may be used. (Note, this may not be the IMS defined type.)

-For TOTAL:
Types S, N, C and P may be used.

Note: TOTAL does not define data types, only storage. These should be obtained from actual program usage.

UNION_VALUE

A value that exists in a specific data field in every record instance for which an EC-RT mapping is valid. This only exists for record types that result from relational unions of entity classes.

VIEW_ID

A surrogate entity class identification is a code that is assigned to uniquely identify a surrogate entity class.

VIEW_NO

Role name for Object Number, introduced in the User View entity class.